



WYDZIAŁ  
ELEKTROTECHNIKI  
I AUTOMATYKI

Imię i nazwisko studenta: Emil Pelak  
Nr albumu: 182651  
Poziom kształcenia: Studia pierwszego stopnia  
Forma studiów: niestacjonarne  
Kierunek studiów: Elektrotechnika  
Specjalność/profil: -

## **PRACA DYPLOMOWA INŻYNIERSKA**

Tytuł pracy w języku polskim: Projekt stacji monitorującej parametry powietrza.

Tytuł pracy w języku angielskim: Design of an air parameters monitoring station.

Opiekun pracy: dr inż. Artur Cichowski

## **OŚWIADCZENIE dotyczące pracy dyplomowej zatytułowanej: Projekt stacji monitorującej parametry powietrza.**

Imię i nazwisko studenta: Emil Pelak  
Data i miejsce urodzenia:  
17.07.1998, Lublin Nr albumu:  
182651  
Wydział: Wydział Elektrotechniki i Automatyki  
Kierunek: elektrotechnika  
Poziom kształcenia:  
pierwszy Forma studiów:  
niestacjonarne  
Typ pracy: praca dyplomowa inżynierska

Świadomy(a) odpowiedzialności karnej z tytułu naruszenia przepisów ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz. U. z 2019 r. poz. 1231, z późn. zm.) i konsekwencji dyscyplinarnych określonych w ustawie z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (t.j. Dz. U. z 2020 r. poz. 85, z późn. zm.),<sup>1</sup> a także odpowiedzialności cywilnoprawnej oświadczam, że przedkładana praca dyplomowa została opracowana przeze mnie samodzielnie.

Niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem tytułu zawodowego.

Wszystkie informacje umieszczone w ww. pracy dyplomowej, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami zgodnie z art. 34 ustawy o prawie autorskim i prawach pokrewnych.

19.02.2024, Emil Pelak

Data i podpis lub uwierzytelnienie w portalu uczelnianym Moja PG

*\*) Dokument został sporządzony w systemie teleinformatycznym, na podstawie §15 ust. 3b Rozporządzenia MNiSW z dnia 12 maja 2020 r. zmieniającego rozporządzenie w sprawie studiów (Dz.U. z 2020 r. poz. 853). Nie wymaga podpisu ani stempla.*

---

<sup>1</sup> Ustawa z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce:

Art. 312. ust. 3. W przypadku podejrzenia popełnienia przez studenta czynu, o którym mowa w art. 287 ust. 2 pkt 1–5, rektor niezwłocznie poleca przeprowadzenie postępowania wyjaśniającego.

Art. 312. ust. 4. Jeżeli w wyniku postępowania wyjaśniającego zebrany materiał potwierdza popełnienie czynu, o którym mowa w ust. 5, rektor wstrzymuje postępowanie o nadanie tytułu zawodowego do czasu wydania orzeczenia przez komisję dyscyplinarną oraz składa zawiadomienie o podejrzeniu popełnienia przestępstwa.

## STRESZCZENIE

Celem pracy jest zaprojektowanie oraz wykonanie modelu stacji, która monitoruje oraz zbiera w czasie rzeczywistym dane o parametrach, takich jak: temperatura powietrza, ciśnienie atmosferyczne oraz wilgotność powietrza. Urządzenie zostało oparte na mikrokontrolerze z rodziny AVR – ATmega32A. Zaprojektowane urządzenie przetwarza dane z wielofunkcyjnego czujnika środowiskowego BME280 oraz z czujnika temperatury DS18B20. Analizowane parametry są prezentowane w sposób czytelny dla użytkownika na wyświetlaczu OLED oraz za pomocą terminala portu szeregowego wbudowanego w środowisko programistyczne. Stosując moduł czasu rzeczywistego, zbierane dane są dokładnie umiejscowione na osi czasu. Taka funkcjonalność umożliwia zapisanie i analizę pomiarów tuż po zakończeniu badań w wybranym środowisku pomiarowym.

Tuż po wstępie i omówieniu celu pracy dyplomowej, w drugim rozdziale został przedstawiony zestaw uruchomieniowy, który posłużył do budowy urządzenia. Omówiono tam część sprzętową, która została wykorzystana do budowy stacji. Przedstawiono sposób podłączenia części sprzętowej oraz jej konfigurację.

W trzecim rozdziale poniższej pracy zaprezentowano model stacji pomiarowej stworzonej na bazie zestawu uruchomieniowego oraz omówiono główne funkcjonalności urządzenia.

W czwartym rozdziale omówiono biblioteki wykorzystane w projekcie oraz przeanalizowano kod źródłowy projektu.

W przedostatnim rozdziale, czyli piątym, omówiono zrealizowane badania, które były wykonywane w różnych warunkach otoczenia. Wyniki oraz wykonane na ich podstawie tabele, wykresy zostały omówione.

W ostatnim rozdziale czyli w „Podsumowaniu” zwięźle przedstawiono efekty działania stacji monitorującej parametry powietrza. Przedstawiono zbiorczy komentarz na temat wykonanych badań parametrów powietrza. Komentarze te odnoszą do osiągnięć opisanych w poprzednich rozdziałach pracy oraz do początkowych założeń tego projektu. Przedstawiono również działania, które umożliwiłyby rozwój bieżącego projektu.

### **Słowa kluczowe:**

2.2 Elektrotechnika, elektronika, inżynieria informatyczna. 2.2.a Elektrotechnika i Elektronika.

## **ABSTRACT**

Bachelor thesis title:

### **Design of an air parameters monitoring station.**

The aim of the work is to design and build a model of a station that monitors and collects data in real time on parameters such as air temperature, atmospheric pressure and air humidity. The device is based on a microcontroller from the AVR family - ATmega32A. The designed device processes data from the BME280 multifunctional environmental sensor and the DS18B20 temperature sensor. The analyzed parameters are presented in a user-readable way on the OLED display and using the serial port terminal built into the programming environment. Using a real-time module, the collected data is precisely located on the timeline. This functionality allows you to save and analyze measurements right after completing the research in the selected measurement environment.

Just after the introduction and discussion of the purpose of the diploma thesis, the second chapter presents the development kit used to build the device. The hardware part that was used to build the station was discussed there. The method of connecting the hardware part and its configuration is presented.

The third chapter of the following work presents the model of the measuring station created on the basis of the development kit and discusses the main functionalities of the device.

The fourth chapter discusses the libraries used in the project and analyzes the project's source code.

The penultimate chapter, i.e. the fifth, discusses the research carried out in various environmental conditions. The results and the table and charts prepared on their basis were discussed.

The last chapter, i.e. "Summary", briefly presents the effects of the operation of the station monitoring air parameters. A collective comment on the air parameters tests performed was presented. These comments refer to the achievements described in the previous chapters of the work and to the initial assumptions of this project. Activities that would enable the development of the current project were also presented.

### **Keywords:**

2.2 Electrical engineering, electronics, IT engineering. 2.2.a Electrical Engineering and Electronics.

## SPIS TREŚCI

SPIS TREŚCI .....	5
WYKAZ WAŻNIEJSZYCH OZNACZEŃ I SKRÓTÓW .....	6
1. WSTĘP I CEL PRACY .....	8
1.1. Wprowadzenie .....	8
1.2. Stacja meteorologiczna – najważniejsze informacje .....	9
1.3. Wpływ podstawowych parametrów powietrza na zdrowie człowieka .....	11
1.4. Cel i zakres pracy .....	13
2. CHARAKTERYSTYKA ZESTAWU URUCHOMIENIOWEGO ATNEL ATB 1.05A ANDROMEDA .....	15
2.1. Prezentacja części sprzętowej i programowej zestawu uruchomieniowego Atnel ATB 1.05a Andromeda .....	16
2.2. Sposób podłączenia części sprzętowej oraz jej konfiguracja .....	21
3. CHARAKTERYSTYKA STACJI MONITORUJĄCEJ PARAMETRY POWIETRZA .....	25
3.1. Zaprezentowanie stacji pomiarowej stworzonej na bazie zestawu uruchomieniowego Atnel ATB 1.05a Andromeda .....	25
3.2. Omówienie funkcjonalności stacji pomiarowej .....	27
4. ANALIZA KODU ŹRÓDŁOWEGO STACJI POMIAROWEJ .....	32
4.1. Wstęp do kodu oraz omówienie bibliotek wykorzystanych w projekcie .....	32
4.2. Omówienie kodu źródłowego projektu .....	33
5. OMÓWIENIE ZREALIZOWANYCH BADAŃ ZA POMOCĄ STACJI POMIAROWEJ .....	46
5.1. Badanie nr 1. Zamarznięta woda w plastikowym kubku o pojemności 250 ml .....	46
5.2. Badanie nr 2. Gorąca woda w plastikowym kubku o pojemności 250 ml .....	48
5.3. Badanie nr 3. Porównanie czujników DS18B20 oraz BME280 pod kątem precyzji pomiaru temperatury. Odległość między sensorami $l = 0,5$ cm .....	51
5.4. Badanie nr 4. Badanie warunków atmosferycznych na zewnątrz budynku .....	53
6. PODSUMOWANIE .....	55
Wykaz literatury .....	56
Zawartość załączonego pliku .....	58
Wykaz rysunków .....	59
Wykaz tabel .....	62
ZAŁĄCZNIK A: Microchip Studio – uruchomienie projektu i omówienie środowiska .....	63
ZAŁĄCZNIK B: Obsługa stacji pomiarowej oraz logowanie danych pomocą komunikacji UART .....	67

## WYKAZ WAŻNIEJSZYCH OZNACZEŃ I SKRÓTÓW

<i>AI</i>	– ang. Artificial Intelligence – sztuczna inteligencja,
<i>BMS</i>	– ang. Building Management System – system zarządzania budynkiem,
<i>Atmel AVR</i>	– typ ośmiobitowych mikrokontrolerów produkowanych przez firmę Atmel,
<i>Arduino</i>	– platforma programistyczna dla systemów wbudowanych,
<i>STM32</i>	– rodzina 32-bitowych mikrokontrolerów w układach scalonych produkowanych przez francusko-włoską firmę STMicroelectronics,
<i>OLED</i>	– ang. Organic Light Emitting Diode - organiczna dioda elektroluminescencyjna. OLED oznacza również klasę wyświetlaczy graficznych opartych technologii OLED,
<i>RS232</i>	– standard szeregowej transmisji danych między urządzeniami elektronicznymi,
<i>RTC</i>	– ang. real-time clock - zegar czasu rzeczywistego,
<i>I<sup>2</sup>C</i>	– ang. Inter-Integrated Circuit – interfejs komunikacyjny,
<i>1-Wire</i>	– interfejs elektroniczny, jak również i protokół komunikacyjny pomiędzy urządzeniami,
<i>RC5</i>	– protokół komunikacji bezprzewodowej za pomocą podczerwieni, który został stworzony przez firmę Philips w latach 80-tych,
<i>UART</i>	– ang. universal asynchronous receiver-transmitter- uniwersalny asynchroniczny nadajnik-odbiornik,
<i>USART</i>	– ang. universal synchronous/asynchronous receiver/transmitter - uniwersalny synchroniczny i asynchroniczny nadajnik-odbiornik,
<i>PCB</i>	– ang. printed circuit board - płyta drukowana,
<i>USB</i>	– ang. Universal Serial Bus - uniwersalna magistrala szeregową,
<i>Hub USB</i>	– rozdzielacz USB jest urządzeniem, które stosuje się, gdy podstawowa liczba portów USB w danym sprzęcie jest niewystarczająca,
<i>V</i>	– Wolt, jednostka potencjału elektrycznego, napięcia elektrycznego i siły elektromotorycznej,
<i>A</i>	– Amper, jednostka natężenia prądu elektrycznego,
<i>GND</i>	– ang. ground. Punkt w obwodzie elektrycznym, którego potencjał jest potencjałem odniesienia przy ustalaniu napięcia wszystkich innych punktów obwodu,
<i>ISP</i>	– ang. In-System Programming. Rozwiązanie konstrukcyjne stosowane w mikrokontrolerach, umożliwiające zaprogramowanie układu scalonego bez jego demontażu z urządzenia, w którym pracuje,
<i>KANDA</i>	– standard złącza przeznaczonego dla mikrokontrolerów z rodziny AVR,
<i>RS485</i>	– standard telekomunikacyjny opisujący sposób połączenia urządzeń typu sterownik oraz odbiornik w systemach transmisji szeregowej,
<i>V-USB</i>	– programowa implementacja urządzenia USB o niskiej prędkości dla mikrokontrolerów AVR firmy Atmel, która umożliwia zbudowanie

	sprzętu USB z niemal każdym mikrokontrolerem AVR,
<i>LED</i>	– ang. Light-Emitting Diode – dioda emitująca światło,
<i>Hz</i>	– Herc, jednostka miary częstotliwości w układzie SI,
$\Omega$	– Om, jednostka rezystancji w układzie SI,
<i>Przetwornik ADC (A/D)</i>	– przetwornik analogowo-cyfrowy. Układ służący do zamiany sygnału analogowego na sygnał cyfrowy,
<i>F</i>	– Farad, jednostka pojemności elektrycznej w układzie SI,
<i>PWM</i>	– modulacja szerokości impulsów,
<i>Fusebity</i>	– bity konfiguracyjne, służące do konfiguracji parametrów pracy mikrokontrolera,
<i>Firmware</i>	– oprogramowanie zainstalowane na stałe w urządzeniu, zapewniające podstawowe procedury jego obsługi,
<i>IDE</i>	– ang. Integrated Development Environment – zintegrowane środowisko programistyczne,
<i>CMOS</i>	– ang. Complementary Metal-Oxide-Semiconductor - technologia wytwarzania układów scalonych,
<i>RISC</i>	– ang. Reduced Instruction Set Computing - typ architektury zestawu instrukcji procesora,
<i>b</i>	– bit, jest to najmniejsza ilość informacji potrzebna do określenia, który z dwóch równie prawdopodobnych stanów przyjął układ,
<i>B</i>	– bajt, jest to najmniejsza adresowalna jednostka informacji pamięci komputerowej, współcześnie składająca się zawsze z 8 bitów,
<i>Timer</i>	– układ liczący, w mikrokontrolerach 8-bitowych najczęściej o rozdzielczości 8 lub 16 bitów,
<i>RH</i>	– ang. Relative Humidity. Dopuszczalna względna wilgotność otoczenia, której wartość wyrażona jest w procentach od 0 do 100 %,
<i>AGND</i>	– masa części analogowej,
<i>DGND</i>	– masa części cyfrowej,
<i>RX</i>	– ang. Receiver – odbiornik,
<i>TX</i>	– ang. Transmitter – nadajnik,
<i>RAM</i>	– ang. random-access memory - pamięć o dostępie swobodnym,
<i>CTC</i>	– ang. Clear Timer on Compare Match – tryb pracy timera sprzętowego,
<i>CPU</i>	– ang. central processing unit – procesor.

# 1. WSTĘP I CEL PRACY

## 1.1. Wprowadzenie

W ostatnich latach można zaobserwować bardzo dynamiczny rozwój elektroniki. Jest to związane z coraz większym popytem oraz podażą w tej dziedzinie gospodarki oraz przemysłu. Otaczający nas świat rozwija się coraz szybciej, a do światowej gospodarki dołączają nowe państwa, które chcą rozwijać się w dziedzinie nowych technologii. Nowoczesna elektronika otacza nas z każdej strony, a dostępne urządzenia na rynku mają w swojej nazwie dopisek „Smart” lub „wyposażone w sztuczną inteligencję – AI”. Często są to tylko zabiegi marketingowe, ale ciężko tego nie zauważyć, że wszystkie nowe urządzenia wyposażone są w nowoczesne mikroprocesory, których cena jest coraz niższa, a implementacja jest coraz łatwiejsza. Taka sytuacja na globalnych rynkach sprzyja powstawaniu nowych firm, które projektują elektronikę, rozwijają oprogramowanie dla systemów wbudowanych, a ponadto testują oprogramowanie dla uznanych światowych marek.

W Internecie oraz w dużych marketach elektronicznych, możemy się spotkać z ogromnym wyborem sprzętu RTV oraz AGD. Są to najczęściej wybierane przez nas urządzenia. Wybieramy je, aby wyposażać domy, biura oraz inne miejsca pracy. Nie trudno się zgodzić ze stwierdzeniem, że jednym z najbardziej rozpowszechnionych urządzeń jest smartfon, czyli przenośny komputer w kieszeni użytkownika – mamy go praktycznie zawsze przy sobie. Należy jednak wziąć pod uwagę, że elektronika oraz systemy komputerowe przetwarzające dane są w bardzo dużej skali używane w innych branżach takich jak:

- inteligentne miasta;
- inteligentne fabryki;
- systemy zarządzania budynkiem;
- branża medyczna;
- automatyka i robotyka;
- przemysł motoryzacyjny.

Wszystkie wyżej wymienione branże charakteryzują się przetwarzaniem oraz gromadzeniem danych o otaczającym nas otoczeniu oraz o nas samych. Temat mojej pracy dyplomowej, czyli „Projekt stacji monitorującej parametry powietrza” ma w tym kontekście swoje uzasadnienie, ponieważ w każdym z tych przypadków niezwykle ważne jest kontrolowanie warunków pracy, warunków otoczenia oraz bezpieczeństwa użytkownika. Analiza i przetwarzanie informacji o parametrach powietrza jest bardzo ważna w profesjonalnych oraz amatorskich zastosowaniach. Na co dzień mamy do czynienia z dużą ilością urządzeń pomiarowych oraz sterujących, które wykorzystują te zgromadzone dane. Mówiąc o zastosowaniach profesjonalnych, należy wspomnieć o systemach zarządzania budynkiem (ang. BMS – Building Management System). Jest to branża, w której ważną rolę odgrywa analiza parametrów powietrza. Systemy te są szeroko stosowane między innymi w inteligentnych budynkach, a swoje działanie opierają na pomiarach właściwości otoczenia. Monitorując wartości parametrów powietrza takich jak.: temperatura, ciśnienie atmosferyczne, wilgotność, stężenie niektórych gazów oraz pyłów



zawieszonych, są w stanie odpowiednio sterować wentylacją, klimatyzacją oraz ogrzewaniem. Dzięki analizie wcześniej wymienionych parametrów, możliwa jest również integracja innych systemów występujących na obiekcie, takich jak:

- podsystem zasilania oraz sterowania energią elektryczną;
- podsystem sterowania komfortem.

Opisane systemy są szeroko rozpowszechnione w szpitalach, osiedlach mieszkaniowych, domkach jednorodzinnych, centrach danych, centrach handlowych, biurach oraz w hotelach. Dzięki integracji poszczególnych systemów, możliwa jest obsługa interfejsu graficznego, w którym możemy kontrolować parametry pracy oraz zmieniać wartości nastawne. Obecność takiego rozwiązania pozwala zaoszczędzić czas, a przede wszystkim pieniądze.

Tematyka monitorowania stanu atmosfery nabiera wyjątkowego znaczenia w dobie walki z wszechobecnym smogiem w miastach oraz w obszarach silnie zurbanizowanych. Na rynku można obecnie zakupić rozwiązania dla amatorów, którzy chcą na bieżąco monitorować warunki panujące wewnątrz oraz na zewnątrz badanego środowiska, np. temperatura powietrza, ciśnienie atmosferyczne, wilgotność i zanieczyszczenie powietrza. Takim rozwiązaniem może być stacja pogodowa, lecz znacznie częściej są to złożone systemy, których zadaniem jest oczyszczanie, osuszanie i nawilżanie powietrza. Powietrze utrzymane na optymalnym poziomie to najbardziej pożądana sytuacja w domach.

W związku z rozpowszechnieniem elektroniki oraz gotowych komponentów na rynku, można zakupić gotowe rozwiązanie stacji pogodowej lub stworzyć własne rozwiązanie. Branża elektrotechniczna to szerokie spektrum wiedzy oraz umiejętności, które powinien posiadać inżynier elektrotechniki. W dobie dynamicznie zmieniającego się świata, zdolność tworzenia oprogramowania dla mikrokontrolerów jest bardzo istotna. W zależności od wybranej platformy (np. Atmel AVR, Arduino, STM32) dla projektu, uruchomienie w pełni działającego urządzenia zgodnie z naszymi wytycznymi oraz oczekiwaniami, może wymagać różnego nakładu czasu, pracy oraz umiejętności.

## **1.2. Stacja meteorologiczna – najważniejsze informacje**

Stacja meteorologiczna – jest to zazwyczaj jednostka służby meteorologicznej [1], np. Instytutu Meteorologii i Gospodarki Wodnej (czyli IMGW). Poza obserwacją składników pogody za pomocą różnego rodzaju aparatury badawczej, zadaniem stacji jest systematyczne dostarczanie pomiarów w czasie rzeczywistym do właściwej jednostki badawczej.

Istnieje możliwość wyboru programu obserwacyjnego i metod pomiaru. Jeśli jest taka potrzeba, to stacja może równie dobrze działać jako jednostka zapisująca dane z pomiarów bez ich bezpośredniej transmisji do użytkownika końcowego, wykorzystując np. kartę pamięci lub dysk twardy do zapisu danych. Dzięki zastosowaniu takich stacji pomiarowych, możliwa jest obserwacja podstawowych elementów oraz zjawisk meteorologicznych, takich jak:

- temperatura powietrza, czyli jeden z podstawowych elementów meteorologicznych, określający stan cieplny atmosfery [2] (pomiaru są dokonywane na wysokości 2 metrów nad gruntem za pomocą termometru lub czujnika temperatury. Klatka meteorologiczna

osłania element pomiarowy przed bezpośrednim promieniowaniem. Pomiar temperatury minimalnej oraz maksymalnej jest wykonywany za pomocą dwóch termometrów (minimalnym oraz maksymalnym). Zazwyczaj wynik jest podawany w stopniach Celsjusza [ $^{\circ}\text{C}$ ], Fahrenheita [ $^{\circ}\text{F}$ ] lub w kelwinach [ $\text{K}$ ]. W przypadku analizy danych, ważnym parametrem jest amplituda temperatury powietrza, czyli różnica między najwyższą, a najniższą temperaturą w danym okresie;

- ciśnienie atmosferyczne, które informuje nas, jaki nacisk wywiera powietrze na jednostkę powierzchni ciała (np. na  $1\text{m}^2$ ) [3]. Jednostką miary ciśnienia jest paskal [ $\text{Pa}$ ]. Parametr ten zmienia się w zależności od wysokości nad poziomem morza, szerokości geograficznej oraz temperatury powietrza. Ciśnienie powietrza spada wraz wysokością, a zależność ta jest w przybliżeniu malejąca wykładniczo [4]. Wyższa temperatura powietrza sprawia, że powietrze jest lżejsze, czego skutkiem jest niższe ciśnienie atmosferyczne i mniejsza zawartość tlenu w powietrzu. Do pomiaru ciśnienia wykorzystujemy barometr;
- wilgotność powietrza - jest to zawartość pary wodnej w powietrzu, a wyniki pomiarów podajemy w procentach % [5]. Parametr ten jest ważnym czynnikiem określającym stan atmosfery oraz jednym z głównych wskaźników pogody. Analizując zawartość pary wodnej w powietrzu, można określić prawdopodobieństwo wystąpienia zjawisk atmosferycznych, takich jak: opady atmosferyczne, rosa, mgła lub przymrozki. Wilgotność powietrza można zmierzyć za pomocą higrometru oraz psychrometru;
- prędkość oraz kierunek wiatru, który jest możliwy do określenia znając rozkład obszarów o niskim i wysokim ciśnieniu. Wiatr wieje zawsze od wyżu do niżu, a jego kierunek jest podawany w zależności od strony, z której wieje. Prędkość wiatru jest proporcjonalna do różnicy ciśnień. Wartość prędkości oraz kierunku wiatru dokonuje się za pomocą wiatromierza [6];
- zachmurzenie, które jest określane w skali od 0 do 8 na podstawie stopnia pokrycia nieba przez chmury. Ważna jest również obserwacja typów chmur oraz kolejność ich występowania [7];
- opady atmosferyczne, które są ciekłymi lub stałymi produktami kondensacji pary wodnej spadającymi z chmur na powierzchnię Ziemi. Wśród opadów zalicza się: mżawkę, deszcz, grad, śnieg oraz krupy śnieżne. Wyniki pomiarów podajemy w milimetrach „słupa” wody, która spada na daną powierzchnię. 1 mm opadów to 1 litr opadów przypadający na  $1\text{ m}^2$  podłoża [8];
- usłonecznienie – podawany w godzinach (h) parametr pogody informuje nas, przez jaki czas na określone miejsce padają promienie słoneczne. Z naturalnych względów, usłonecznienie jest większe latem, natomiast mniejsze zimą [9];

Zaawansowane urządzenia pomiarowe są rozmieszczone na całej kuli ziemskiej. Można je znaleźć w takich miejscach jak:

- lądy;
- oceany;

- na zakotwiczonych lub pływających statkach;
- na obszarach trudno dostępnych dla człowieka, takich jak: Arktyka, Antarktyda oraz gorące pustynie.

Można wyróżnić kilka typów stacji pomiarowych, wykorzystywanych w zależności od wymagań środowiskowych oraz rodzaju pomiarów, na przykład:

- synoptyczne - dostarczają dane dla map synoptycznych, które graficznie przedstawiają obecny stan pogody na większym obszarze, np. Polski;
- aerologiczne – zajmują się badaniem procesów i zjawisk zachodzących w górnych warstwach atmosfery. Pomiary te są realizowane za pomocą przyrządów umieszczonych na wieżach, podwieszonych pod balonami oraz zainstalowanymi na dronie bądź w samolocie [10];
- klimatologiczne – za ich pomocą uzyskujemy dane na temat procesów fizycznych, które kształtują klimat;
- agrometeorologiczne – realizowane obserwacje elementów metrologicznych są powiązane z przebiegiem wegetacji i stanem roślin uprawnych oraz biorą pod uwagę wpływ pogody na prace gospodarskie w rolnictwie i leśnictwie [11];
- lotniczo-meteorologiczne – dostarczają bardzo ważnych pomiarów dla ruchu lotniczego;
- stacja aktynometryczna, która dostarcza dane o promieniowaniu Słońca, Ziemi i atmosfery. Jej pomiary dostarczają informacji o promieniowaniu bezpośrednim, rozproszonym i pochłoniętym w atmosferze, a także wyznaczaniem promieniowania efektywnego oraz bilansu promieniowania Ziemi i atmosfery [12].

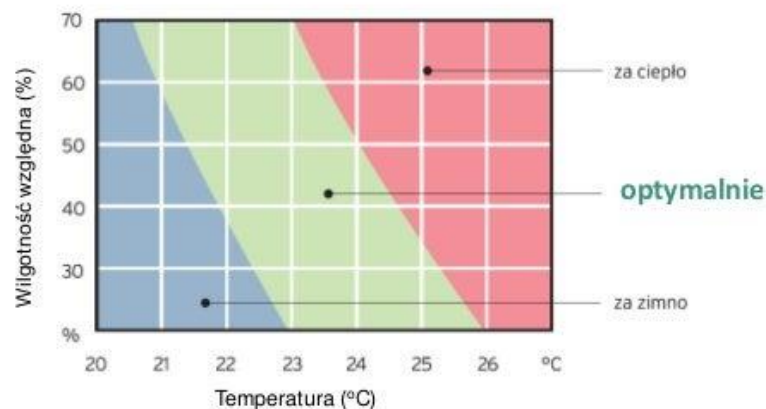
### **1.3. Wpływ podstawowych parametrów powietrza na zdrowie człowieka**

Temperatura oraz wilgotność powietrza, to parametry które wpływają na mikroklimat w pomieszczeniu, w którym się znajdujemy. Wskaźniki te mają dla nas ważne znaczenie, gdy wychodzimy na zewnątrz lub gdy chcemy zaplanować nadchodące dni. Niewątpliwie jest również wpływ ciśnienia atmosferycznego, które w mniejszym lub większym stopniu wpływa na samopoczucie w ciągu dnia.

W tym podrozdziale chciałbym omówić wpływ tych parametrów na zdrowie oraz samopoczucie człowieka, ponieważ ta wiedza będzie przydatna podczas omawiania kolejnych rozdziałów.

Należy pamiętać, że temperatura oraz wilgotność powietrza są ze sobą powiązane i zależne od siebie. Im wyższa temperatura otoczenia, tym powinno ono mieć niższą wilgotność, aby uzyskać odpowiedni komfort cieplny dla człowieka. Połączenie wysokiej temperatury oraz wilgotności prowadzi do rozwoju grzybów oraz pleśni [13].

- Wilgotność względna powietrza pozostaje w związku z jego temperaturą. Im wyższa temperatura powietrza, tym powinno ono mieć niższą wilgotność dla uzyskania odpowiedniego komfortu cieplnego człowieka.



Źródło: HEA Fachverband für Energie-Marketing und -Anwendung e.V. beim VDEW.

Vaillant

8

Rys. 1.1. Optimalny zakres wilgotności względnej powietrza, a komfort cieplny powietrza [13]

Na rysunku 1.1. znajduje się wykres, który przedstawia zależność między temperaturą, a wilgotnością w celu utrzymania mikroklimatu w budynku. Na jego podstawie można wywnioskować, że zalecana temperatura w pomieszczeniu mieszkalnym powinna wynosić od 21 °C przy wilgotności 60 %, do 24,5 °C przy wilgotności 40 %. Parametry te są związane z optymalną wilgotnością dla człowieka. Warto wziąć pod uwagę, że zalecana temperatura w sypialni powinna oscylować w okolicach 18 °C.

Konsekwencje zbyt suchego oraz wilgotnego powietrza mają negatywny wpływ na zdrowie [13]. Ciepłe i wilgotne warunki w naszym domu to idealne warunki dla rozwoju wirusów, bakterii, grzybów, roztoczy oraz pleśni, które są niczym innym jak szkodliwymi patogenami. Dlatego tak bardzo ważne jest, aby wentylować pomieszczenia, w których przebywamy.

Natomiast zbyt suche powietrze może spowodować gorszą pracę systemu immunologicznego oraz obniżyć naszą odporność na infekcje. Takie warunki powodują wysuszenie skóry oraz błon śluzowych.

Chciałbym również omówić wpływ ciśnienia atmosferycznego na zdrowie. To właśnie jego zmiany mogą powodować zmiany samopoczucia, nastroju i nie tylko [14]. Ciśnienie atmosferyczne na nizinach, będzie wyższe niż w górach. Dzieje się tak, ponieważ słup powietrza atmosferycznego, który naciska na powierzchnię Ziemi, ma różną wysokość. Prawidłowa wartość ciśnienia atmosferycznego wynosi 1013,25 hPa. Niekorzystne zmiany ciśnienia atmosferycznego wynoszą [14]:

- 3 hPa w ciągu 3 godzin;
- powyżej 8 hPa w ciągu doby;

Osobami najbardziej narażonymi na zmiany tego wskaźnika są meteopaci. Odsetek tych ludzi w porównaniu do zdrowego społeczeństwa wynosi od 50 do 70 %, jednak jego wartość ciągle rośnie (najbardziej narażoną grupą są kobiety, osoby przepracowane oraz żyjące w dużym stresie). Meteopaci podczas zmiany pogody mogą odczuwać zawroty głowy, bóle głowy, nagłe skoki ciśnienia krwi, nadwrażliwość na światło, nudności, zmęczenie oraz przygnębienie [15].

Niskie oraz wysokie ciśnienie może mieć na nas negatywny wpływ. Spadek poniżej prawidłowego ciśnienia powietrza, czyli 1013,25 hPa, określamy jako niskie ciśnienie atmosferyczne. Natomiast wzrost powyżej wartości prawidłowej, nazywamy wysokim ciśnieniem atmosferycznym.

Niskie ciśnienie powietrza powoduje, że czujemy się senni oraz zmęczeni. Zmiana ta jest szczególnie niebezpieczna dla osób, które zmagają się z chorobami takimi jak: padaczka, depresja oraz nerwica. Natomiast wysokie ciśnienie może mieć wpływ na ból głowy, problem z koncentracją, senność, rozdrażnienie, zmęczenie oraz problemy z oddychaniem. Aby zneutralizować skutki zmiany ciśnienia atmosferycznego na samopoczucie, warto przeznaczyć większą ilość czasu sen, odpoczynek oraz aktywność fizyczną.

#### **1.4. Cel i zakres pracy**

Celem tej pracy inżynierskiej jest stworzenie modelu stacji monitorującej oraz zbierającej w czasie rzeczywistym dane o najważniejszych parametrach powietrza, takich jak: temperatura, ciśnienie oraz wilgotność. Właściwości powietrza według założeń projektu, mogą być badane w miejscu gdzie przebywamy np. pokój dzienny lub na zewnątrz budynku.

Urządzenie będzie wykorzystywane w zastosowaniach domowych (amatorskich) oraz ma ułatwić bieżącą analizę parametrów, które znacząco wpływają na nasz komfort. Monitorowanie oraz utrzymanie podstawowych parametrów powietrza w normie jest bardzo ważne dla zdrowia.

Analizowane wskaźniki w naszym otoczeniu będą prezentowane w sposób czytelny dla obserwatora na wyświetlaczu OLED. Parametry te będą również wysyłane za pomocą terminala portu szeregowego wbudowanego w środowisko programistyczne. Dzięki połączeniu komunikacji RS232 oraz modułu RTC, zbierane dane będą mogły zostać zapisane i przeanalizowane po zakończeniu obserwacji otoczenia.

Zakres pracy obejmują następujące punkty:

- implementacja programowej obsługi wielofunkcyjnego czujnika środowiskowego BME280 poprzez magistralę I<sup>2</sup>C;
- implementacja programowej obsługi czujnika temperatury DS18B20 poprzez interfejs 1-Wire;
- implementacja programowej obsługi układu RTC poprzez magistralę I<sup>2</sup>C;
- zastosowanie wyświetlacza siedmiosegmentowego do wyświetlania aktualnego czasu;
- opracowanie i implementacja programowej obsługi interfejsu użytkownika z wykorzystaniem wyświetlacza OLED;
- wykorzystanie zielonej, migającej diody LED do sygnalizacji pracy urządzenia;
- implementacja programowej obsługi modelu stacji za pomocą fizycznych przycisków;

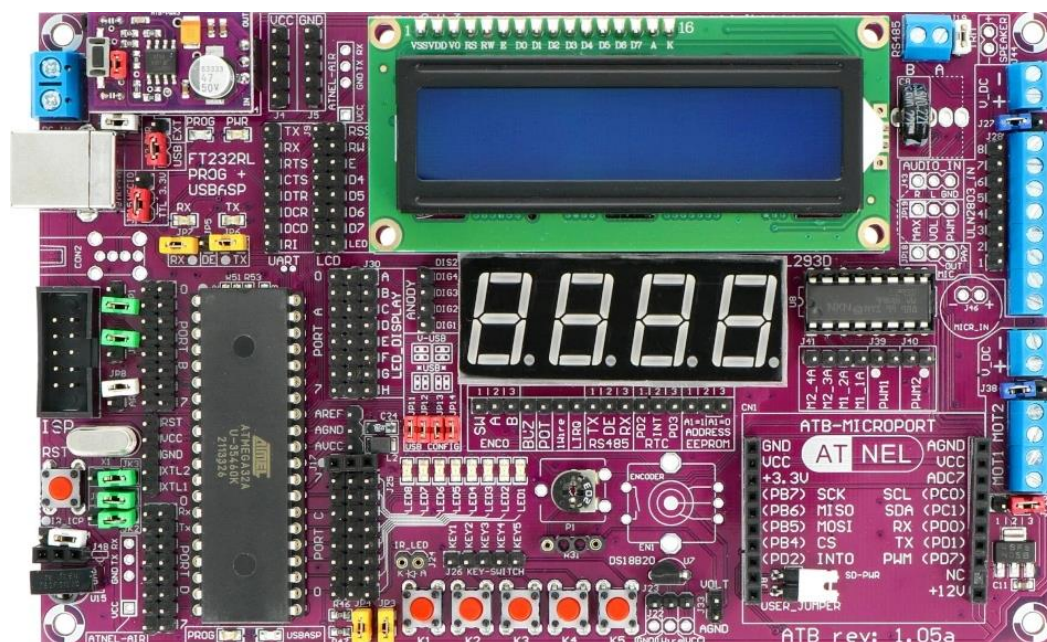
- implementacja programowej obsługi urządzenia za pomocą pilota podczerwieni w standardzie RC5;
- implementacja programowej obsługi dwustronnej komunikacji urządzenia oraz komputera za pomocą interfejsu USART;
- wykorzystanie interfejsu USART oraz terminala komputerowego w celu wizualizacji danych zbieranych przez urządzenie;
- przeprowadzenie badań, które zostaną wykonane w różnych interwałach oraz środowiskach. Przetworzenie wyników oraz wykonanie na ich podstawie tabel oraz wykresów.

## 2. CHARAKTERYSTYKA ZESTAWU URUCHOMIENIOWEGO ATNEL ATB 1.05A ANDROMEDA

Do stworzenia stacji monitorującej parametry powietrza, wykorzystano zestaw uruchomieniowy (ewaluacyjny), który został zaprojektowany i wyprodukowany przez polską firmę *Atnel* [16]. Wybór tej platformy sprzętowej wynika z tego, że posiada ona wiele zalet, takich jak:

- obecność mikroprocesora ATmega32A oraz możliwość realizacji projektów w języku C;
- obecność wielu modułów na jednej płytce PCB, umożliwiającej wykonanie zróżnicowanych ćwiczeń oraz projektów;
- prosta rekonfiguracja urządzenia (między innymi zmiana pinów);
- możliwość przenoszenia urządzenia bez obaw o uszkodzenie połączeń między podzespołami oraz modułami na PCB;
- wysoka jakość wykonania oraz opis poszczególnych modułów na PCB;
- duże wsparcie merytoryczne i dydaktyczne dla zestawu w postaci książek oraz poradników wideo;
- obecność wbudowanego programatora ATB-USBasp oraz zapasowego programatora ATB-FT232RL;
- możliwość podłączenia zewnętrznego programatora poprzez złącze KANDA;
- możliwość zewnętrznego zasilania zestawu przez +12 V lub kabel USB;
- możliwość wyboru napięcia na zestawie uruchomieniowym między 3,3 V oraz 5 V;
- 4-portowy Hub USB;

Ze względu na te zalety, urządzenie zostało w pełni skonfigurowane dla tej platformy, która została przedstawiona na rysunku 2.1.



Rys. 2.1. Widok ogólny zestawu uruchomieniowego ATB 1.05a Andromeda [17]

## **2.1. Prezentacja części sprzętowej i programowej zestawu uruchomieniowego Atnel ATB**

### **1.05a Andromeda**

Niewielki rozmiar zestawu uruchomieniowego daje bardzo duże możliwości programistyczne. Obecność wielu zintegrowanych modułów umożliwiła mi zrealizowanie mojego projektu w przejrzysty oraz solidny sposób, bez obaw o problemy z połączeniami oraz odporność na przeniesienie stacji monitorującej parametry powietrza.

Zestaw Atnel ATB 1.05a Andromeda posiada następujące elementy elektroniczne [18] :

- mikrokontroler ATmega32A;
- programator sprzętowy „ATB-USBasp”, wykorzystujący rozwiązanie konstrukcyjne ISP do programowania mikrokontrolera;
- przetwornica ATB-PWR3 z możliwością wyboru zasilania +3,3 V oraz +5 V. Podczas używania domyślnie zainstalowanej przetwornicy ATB-PWR3, do zestawu można doprowadzić napięcie stałe o wartości od +8 V do +40 V. Zalecana wydajność prądowa używanego zasilacza powinna wynosić od 1 A do 1,5 A;
- przejściówka USB/RS232 (układ FT232RL), która umożliwia komunikację poprzez RS232 oraz przez RS485 dzięki wykorzystaniu układu MAX485);
- wymienny rezonator kwarcowy z podstawką (domyślnie 11,0592 MHz);
- przycisk RESET (jego wciśnięcie zwiera bezpośrednio linię RESET mikrokontrolera do GND);
- złącze zewnętrznego programatora typu KANDA;
- sprzętowo przygotowany układ V-USB, który można za pomocą dwóch przewodów podłączyć do mikrokontrolera;
- 4-portowy HUB-USB, którego pierwszy port obsługuje programator ATB-USBasp, a drugi port przejściówkę USB/RS232 (FT232R). Trzeci port jest dostępny dla użytkownika z możliwością jego konfiguracji w tryb V-USB. Dużą zaletą jest to, że czwarty wolny port jest dostępny na złącz kątowym USB-A z możliwością podłączenia na przykład zewnętrznego programatora;
- 8 zielonych diod LED, które mogą być sterowane z poziomu mikrokontrolera jako wyjście;
- odbiornik promieniowania podczerwonego dla częstotliwości nośnej 36 kHz;
- dioda nadawcza promieniowania podczerwonego;
- wyświetlacz alfanumeryczny LCD w kolorze niebieskim;
- Wyświetlacz 7-segmentowy LED w kolorze czerwonym, posiadający cztery cyfry oraz cztery kropki;
- 5 przycisków typu tact-switch, które mogą być sterowane z poziomu mikrokontrolera jako wejście;
- potencjometr 20 k $\Omega$ , stosowany w układzie wejściowym przetwornika ADC;
- cyfrowy czujnik temperatury DS18B20, który współpracuje z protokołem 1-Wire;
- zewnętrzna pamięć EEPROM AT24c04 podłączona jest do magistrali I<sup>2</sup>C;
- układ RTC DS1337+ (możliwość osadzenia: PCF8583, PCF8563, DS1307+);



- odpowiednik układu MAX485 (układ SN75176), wykorzystywany dla komunikacji RS485;
- kondensator żelowy o pojemności 0,22 F, pełniący funkcję podtrzymania zasilania dla układu RTC;
- driver mocy ULN2803, posiadający osiem niezależnych kanałów sterowalnych bezpośrednio z mikrokontrolera. Można go użyć do sterowania obciążeniami do 500 mA, takimi jak: silnik krokowy unipolarny oraz przekaźnik;
- złącze ATB-Microport, czyli dwa rzędy złączy żeńskich 10-cio pinowych z wyraźnym opisem wyprowadzonych sygnałów. Dostępne jest tam między innymi zasilanie oraz magistrale takie jak: SPI, I2C oraz UART;
- przetwornica obniżająca napięcie kontrastu do LCD - ICL7660 wraz z kondensatorami;
- wzmacniacz mocy audio TDA7052A z regulacją głośności - PWM;
- wzmacniacz mikrofonowy LM358;
- gniazda ATNEL-AIR, przystosowane dla modułów takich jak: Bluetooth (ATB-BTM-222), Wi-Fi (ATNEL-WIFI232-T) oraz radiomodem (HM-TRP);
- enkoder 24 impulsy (kroki);
- 2 układy optoizolatorów LTV355T wraz ze złączami;
- brzęczyk;

Dla tej płytki rozwojowej dostępne są oprogramowania producenta, które mogą pomóc w pierwszych krokach podczas konfiguracji zestawu uruchomieniowego. W oryginalnej instrukcji dla tego zestawu (umieszczonej w wykazie literatury w punkcie numer 15), zostały przedstawione następujące programy:

- MkAvrCalculator (służący do obliczania fusebitów oraz do programowania mikrokontrolerów);
- MkBootloader (pozwalający na dokonywanie zmian firmware'u w mikrokontrolerach AVR)

Wbudowany programator Atmel ATB-USBasp umożliwia tworzenie oprogramowania wbudowanego z wykorzystaniem zintegrowanego środowiska programistycznego Eclipse.

Do budowy prototypu stacji monitorującej wykorzystano moduły elektroniczne dostępne wraz z zestawem uruchomieniowym oraz komponenty, które zostały podłączone dodatkowo. Lista wykorzystanych komponentów elektronicznych oraz ich opis/specyfikacja techniczna znajduje się w poniższej tabeli 2.1. Sposób podłączenia omawianych modułów elektronicznych zostanie przedstawione w kolejnym podrozdziale.

**Tabela 2.1.** Lista wykorzystanych komponentów elektronicznych oraz jego opis/specyfikacja techniczna

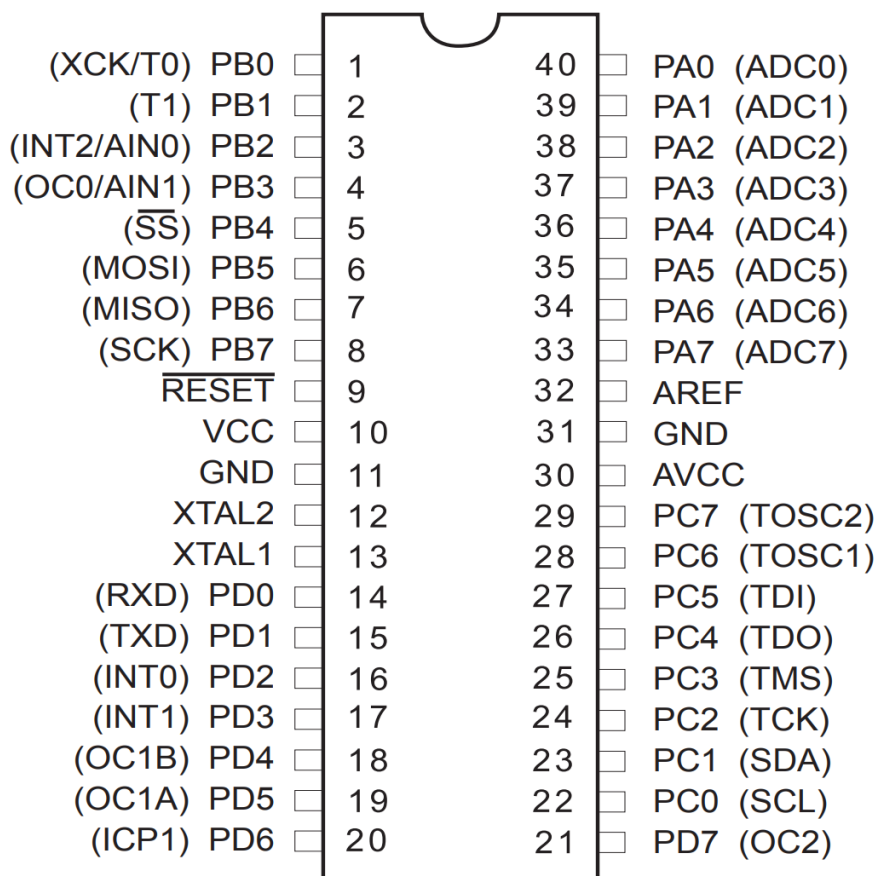
Komponent elektroniczny	Opis/specyfikacja techniczna
ATmega32A	<p>ATmega32A to 8-bitowy mikrokontroler CMOS o niskim poborze mocy, oparty na ulepszonej architekturze RISC AVR [19].</p> <ul style="list-style-type: none"><li>• rdzeń: AVR;</li><li>• napięcie zasilania: <math>2.7 \div 5.5</math> V;</li><li>• obudowa: DIP40;</li><li>• temperatura pracy: <math>-55 \div 125</math> °C;</li><li>• maksymalne taktowanie: 16 MHz;</li><li>• pamięć Flash: 32 kB;</li><li>• pamięć SRAM: 2 kB;</li><li>• pamięć EEPROM: 1 kB;</li><li>• liczba GPIO: 32;</li><li>• interfejs JTAG;</li><li>• interfejsy komunikacyjne: 1 × I<sup>2</sup>C, 1 × SPI, 1 × USART;</li><li>• 1 × ADC 10 bit (8 kanałów, pomiar różnicowy, komparator analogowy);</li><li>• 3 timery: 2 × Timer 8-bitowy, 1 × Timer 16-bitowy;</li><li>• 4 kanały PWM;</li><li>• wsparcie dla biblioteki Qtouch.</li></ul>
Rezonator kwarcowy o wartości 11,0592 MHz	Rezonator kwarcowy jest używany jako źródło sygnału zegarowego w mikrokontrolerach. Pozwala on na uzyskanie znacznie stabilniejszej częstotliwości drgań niż wewnętrzny zegar mikrokontrolera.
Kondensator żelowy o pojemności 0,22 F	Kondensator żelowy o dużej pojemności 0,22 F, pełniący funkcję podtrzymania zasilania dla układu RTC.
Układ RTC DS1337+	DS1337 to zegar czasu rzeczywistego oraz kalendarz o niskim poborze mocy z możliwością programowania dwóch alarmów. Układ posiada wyjście programowalne o przebiegu prostokątnym. Adres oraz dane są przesyłane szeregowo poprzez magistralę I <sup>2</sup> C. Zegar/kalendarz wyświetla sekundy, minuty, godziny, dzień tygodnia, datę, miesiąc oraz rok. Data jest automatycznie korygowana dla miesięcy mających mniej niż 31 dni oraz jest uwzględniona korekta dla roku przestępnego [20].
Wyświetlacz OLED	Wyświetlacz OLED wyświetlający znaki w kolorze niebieskim oraz o przekątnej 1,3" i rozdzielczości 128 × 64 pikseli. Kąt widzenia wynosi powyżej 160 °. Ekran jest oparty na sterowniku SH1106 oraz pracuje z napięciami 3,3 V oraz 5 V. Komunikuje się poprzez magistralę I <sup>2</sup> C. Wyświetlacz może pracować w temperaturze od -20 °C do 70 °C. Jego wymiary wynoszą: 35 × 33 mm [21].
Wyświetlacz 7-segmentowy LED w kolorze czerwonym, posiadający cztery cyfry oraz cztery kropki	W zestawie ATB wyświetlacz jest w konfiguracji ze wspólną anodą - osiem segmentów od A do H (H-kropka). Piny związane z segmentami usytuowane są równolegle do pinów PORTA, dzięki czemu można używać standardowych zworek w rastrze 2,54 mm aby podłączyć wybrane segmenty wygodnie i szybko do mikrokontrolera.

Komponent elektroniczny	Opis/specyfikacja techniczna
	Wyprowadzenia wspólnych anod wyświetlaczy oznaczone są od DIG1 do DIG4. Odpowiadają one za kolejne cyfry od lewej do prawej strony wyświetlacza [18].
Czujnik środowiskowy BME280	<p>Czujnik służący do pomiaru temperatury, ciśnienia atmosferycznego oraz wilgotności. Jest on idealny do aplikacji, które wymagają dokładnych pomiarów, takich jak: stacje pogodowe, systemy kontroli warunków panujących w pomieszczeniach zamkniętych itp. [22, 23].</p> <ul style="list-style-type: none"> <li>Napięcie zasilania modułu: 5 V;</li> <li>Wymiary czujnika: 2,5 × 2,5 × 0,93 mm;</li> <li>Wymiary modułu: 15 × 12 mm;</li> <li>Komunikacja z modułem: magistrala I<sup>2</sup>C;</li> <li>przy odczycie wszystkich trzech parametrów z częstotliwością 1 Hz, BME280 pobiera zaledwie 3,6 μA;</li> <li>Jeden otwór montażowy o średnicy 3 mm;</li> <li>Zakresy pomiarowe: <ul style="list-style-type: none"> <li>Temperatura: <ul style="list-style-type: none"> <li>zakres pomiarowy: od -40 do 85 °C;</li> <li>dokładność: ± 1 °C;</li> </ul> </li> <li>Ciśnienie: <ul style="list-style-type: none"> <li>zakres pomiarowy: od 300 do 1100 hPa;</li> <li>dokładność: ± 1 hPa;</li> </ul> </li> <li>Wilgotność: <ul style="list-style-type: none"> <li>zakres pomiarowy: od 10 do 100 % RH;</li> <li>dokładność: ± 3 % RH.</li> </ul> </li> </ul> </li> </ul>
Cyfrowy czujnik temperatury DS18B20	<p>Zapewnia pomiary temperatury w skali Celsjusza z rozdzielczością od 9 do 12 bitów. Posiada on również wbudowaną funkcję alarmu, który jest wyzwalany, gdy temperatura przekroczy wartość progową. Wartość progowa jest ustawiana programowo przez użytkownika oraz przechowywana w pamięci nieulotnej. Czujnik komunikuje się poprzez magistralę 1-Wire, która z definicji wymaga tylko jednej linii danych oraz masy do komunikacji z mikroprocesorem. Sensor może pobierać energię bezpośrednio z linii danych („zasilanie pasożytnicze”), eliminując potrzebę stosowania zewnętrznego źródła zasilania. Każdy czujnik cyfrowy DS18B20 posiada unikalny 64-bitowy kod seryjny, który umożliwia pracę wielu DS18B20 na tej samej magistrali 1-Wire. Dzięki temu można łatwo użyć jednego mikroprocesora do sterowania wieloma urządzeniami [24, 25].</p> <ul style="list-style-type: none"> <li>Napięcie zasilania: od 3,0 V do 5,5 V;</li> <li>Zakres pomiarowy: od -55 °C do 125 °C;</li> <li>Dokładność: +/- 0,5 °C w zakresie -10°C do 85°C</li> <li>Rozdzielczość: od 9 do 12 bitów;</li> <li>Obudowa THT TO92.</li> </ul>
5 przycisków typu	Są one elementem, które mogą inicjować zwarcie lub rozwarcie obwodu

Komponent elektroniczny	Opis/specyfikacja techniczna
tact-switch	elektrycznego. Mogą zostać wykorzystane jako elementy wejściowe mikroprocesora, a za ich pomocą można uruchomić wybrane funkcje.
Przejściówka USB/RS232 (układ FT232RL)	<p>Układ FT232RL pełni cztery funkcje w zestawie ATB [18]:</p> <ul style="list-style-type: none"> <li>• Przejściówka USB/RS232 z możliwością doboru zakresu napięć na wyjściach. Pozwala ona na wymianę danych między komputerem, a mikrokontrolerem. Wszystkie linie portu RS232 są dostępne dla użytkownika;</li> <li>• Przejściówka USB/RS485, która w połączeniu z osadzonym zamiennikiem układu MAX485 w zestawie daje możliwości podłączania się komputerem PC do magistrali przemysłowej RS485 i dokonywanie dowolnych testów za pomocą własnego lub zewnętrznego oprogramowania;</li> <li>• Programator zapasowy w zestawie typu ATB-FT232RL;</li> <li>• Generator TTL na potrzeby taktowania mikrokontrolera.</li> </ul>
Odbiornik promieniowania podczerwonego TSOP31236	<p>Zminiaturyzowany odbiornik przeznaczony do systemów zdalnego sterowania na podczerwień dla częstotliwości nośnej 36 kHz.</p> <p>Demodulowany sygnał wyjściowy może być bezpośrednio dekodowany przez mikroprocesor [26].</p>
Zielona dioda LED	Półprzewodnikowe źródło światła w kolorze zielonym.
Złącze ATB-Microport	<p>Dostępne są wyprowadzenia napięć, takie jak: +12 V, +5 V, +3,3 V oraz masy GND i AGND. Dodatkowo są tam dostępne wyprowadzenia mikrokontrolera ATmega32A, między innymi: PD2 (INT0), PA7 (ADC7), PD7 (PWM) oraz magistrale SPI, I2C oraz UART. Dostępność tych wyprowadzeń ułatwia pracę i zwiększa przejrzystość połączeń na zestawie uruchomieniowym.</p>
MPLAB Snap – programator/debugger dla mikrokontrolerów Microchip	<p>MPLAB Snap In-Circuit Debugger/Programator pozwala na przystępne, szybkie i łatwe programowanie i debugowanie mikrokontrolerów PIC, dsPIC, AVR, DSC, SAM, CEC oraz PIC32 [27]. Na zestawie prototypowym są dostępne dwa programatory, ale nie są w pełni kompatybilne z oprogramowaniem Microchip Studio. Dodatkowo podłączając ten programator poprzez interfejs JTAG do mikrokontrolera, mamy możliwość debugowania programu w środowisku programistycznym. Jest również możliwość, aby podłączyć się z mikrokontrolerem poprzez ISP.</p>

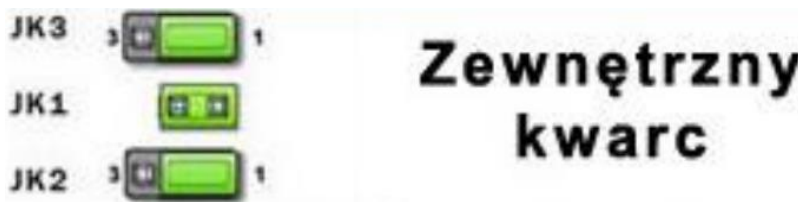
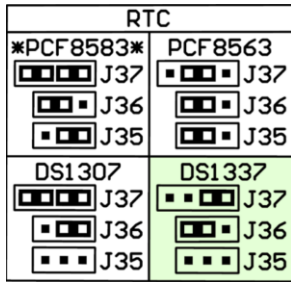
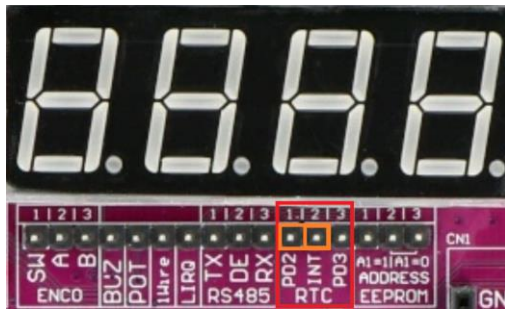
## 2.2. Sposób podłączenia części sprzętowej oraz jej konfiguracja


Na rysunku 2.2. został przedstawiony opis pinów mikrokontrolera ATmega32A z rodziny AVR firmy Atmel w obudowie przewlekanej - DIP40. Na jego podstawie chciałbym przedstawić podłączenie modułów elektronicznych, które zostały opisane w podrozdziale 2.3. W tabeli 2.2. znajdują się wykorzystane komponenty elektroniczne oraz jego połączenie dla zestawu ATB. Część modułów jest ustawiana zworkami na zestawie uruchomieniowym, a szczegółowe schematy są podstępne w instrukcji [18].






Rys. 2.2. Opis wyprowadzeń mikrokontrolera ATmega32A z rodziny AVR firmy Atmel w obudowie przewlekanej - DIP40 [19]

**Tabela 2.2.** Komponenty elektroniczne oraz jego połączenie dla zestawu ATB

Komponent elektroniczny	Połączenie dla zestawu ATB
Rezonator kwarcowy o wartości 11,0592 MHz	<p>Rezonator kwarcowy został umieszczony w dedykowanej podstawie, a zworki JK3, JK1 oraz JK2 zostały ustawione zgodnie z poniższym rysunkiem 2.3.</p>  <p>Rys. 2.3. Ustawienie zworek dla rezonatora kwarcowego w zestawie ATB [18]</p>
Kondensator żelowy o pojemności 0,22 F	Zamontowany domyślnie na PCB.
Układ RTC DS1337+	<p>Zegar czasu rzeczywistego został umieszczony w precyzyjnej podstawie pod wyświetlaczem LCD, a zworki konfiguracyjne J37, J36, J35 zostały ustawione zgodnie z poniższym rysunkiem 2.4. Istnieje możliwość osadzenia czterech układów, a zworkami można szybko rekonfigurować poszczególne układy.</p>  <p>Rys. 2.4. Ustawienie zworek konfiguracyjnych dla RTC w zestawie ATB [18]</p> <p>Dodatkowo wyjście INT modułu RTC (generujące przerwanie co sekundę) zostało podłączone do pinu PD2 (INT0) mikrokontrolera tuż pod wyświetlaczem LED – zgodnie z rysunkiem 2.5.</p>  <p>Rys. 2.5. Podłączenie zegara RTC w zestawie ATB [18]</p> <p>Układ RTC został podłączony do magistrali I<sup>2</sup>C mikrokontrolera. Magistrala I<sup>2</sup>C została podłączona do mikrokontrolera za pomocą zworek JP4, JP3</p>

Komponent elektroniczny	Połączenie dla zestawu ATB
	zgodnie z instrukcją zestawu ATB [18].
Wyświetlacz OLED	Wyświetlacz został wpięty w czteropinowe złącze DIS3 pod wyświetlacz OLED, które znajduje się pod wyświetlaczem LED. Wyświetlacz komunikuje się poprzez magistralę I <sup>2</sup> C.
Wyświetlacz 7-segmentowy LED w kolorze czerwonym, posiadający cztery cyfry oraz cztery kropki	Wyprowadzenia wspólnych anod wyświetlaczy, oznaczone na PCB od DIG1 do DIG4 zostały podłączone do pinów PORTA: PA0, PA1, PA2 oraz PA3. Natomiast wyprowadzenia katod odpowiadających za segmenty od A do H na PCB zostały podłączone do ośmiu pinów PORTB (od PB0 do PB7).
Czujnik środowiskowy BME280	Został on podłączony do złącza ATB-Microport poprzez magistralę I <sup>2</sup> C. Dodatkowo zostało wykorzystane zasilanie 5 V dostępne na złączu.
Cyfrowy czujnik temperatury DS18B20	<p>Czujnik DS18B20 jest zasilany napięciem VCC, które wynosi +5 V. Pin wyjściowy czujnika został podciągnięty do VCC rezystorem 2,2 kΩ. Pin wyjściowy czujnika znajduje się na złączu typu goldpin opisany jako 1Wire – zgodnie z rysunkiem 2.6. Został on podłączony do pinu PD7 mikrokontrolera.</p>  <p>Rys. 2.6. Podłączenie czujnika temperatury w zestawie ATB [18]</p>
5 przycisków typu tact-switch	Zostały one podłączone do pinów PORTA: PA4, PA5, PA6, PA7 oraz PORTC: PC7.
Przełącznik USB/RS232 (układ FT232RL)	Przełącznik jest domyślnie podłączony liniami RX (PD0) i TX (PD1) do mikrokontrolera za pomocą zworek JP6 oraz JP7 zgodnie z instrukcją zestawu ATB [18].
Odbiornik promieniowania podczerwonego TSOP31236	Zworka J14, która jest opisana jako IR_ICP umożliwia podłączenie wyjścia odbiornika do pinu PD6 (ICP1) mikrokontrolera ATmega32A. Zasilanie odbiornika podczerwieni jest filtrowane, a wyjście zostało sprzętowo podciągnięte do VCC.
Zielona dioda LED	Dioda LED1 została podłączona do pinu PC6 mikrokontrolera.
Złącze ATB-Microport	Zamontowane domyślnie na PCB.
MPLAB Snap – programator/debugger dla mikrokontrolerów Microchip	<p>W celu wykorzystania w pełni programatora/debuggera, konieczna była modyfikacja elementów elektronicznych na płycie PCB oraz aktualizacja oprogramowania wbudowanego zgodnie z dokumentacją dostępną w wykazie literatury [28, 29, 30].</p> <p>MPLAB Snap został podłączony do mikrokontrolera za pomocą interfejsu</p>

Komponent elektroniczny	Połączenie dla zestawu ATB																																																					
	<p>JTAG. Można go również podłączyć do mikrokontrolera ATmega32A za pomocą ISP.</p> <p>Na rysunku 2.7. znajduje się opis pinów programatora MPLAB Snap dla interfejsu ISP/JTAG oraz opis pinów mikrokontrolera ATmega32A dla interfejsu JTAG [27].</p> <table><tr><th colspan="3">MPLAB Snap</th><th colspan="2">DEBUG</th><th>ATmega32A</th></tr><tr><th>Connector</th><th>Pin #</th><th>Pin Name</th><th>AVR® JTAG</th><th>AVR ISP(&amp;DW)</th><th>Pin # &amp; Pin Name</th></tr><tr><td rowspan="8"></td><td>1</td><td>TVPP</td><td></td><td></td><td></td></tr><tr><td>2</td><td>TVDD</td><td>VTG</td><td>VTG</td><td>VCC</td></tr><tr><td>3</td><td>GND</td><td>GND</td><td>GND</td><td>GND</td></tr><tr><td>4</td><td>PGD</td><td>TDO</td><td>MISO</td><td>PC4 (TDO)</td></tr><tr><td>5</td><td>PGC</td><td>TCK</td><td>SCK</td><td>PC2 (TCK)</td></tr><tr><td>6</td><td>TAUX</td><td>RESET</td><td>RESET</td><td>RESET</td></tr><tr><td>7</td><td>TTDI</td><td>TDI</td><td>MOSI</td><td>PC5 (TDI)</td></tr><tr><td>8</td><td>TTMS</td><td>TMS</td><td></td><td>PC3 (TMS)</td></tr></table> <p>Rys. 2.7. Opis pinów programatora MPLAB Snap dla interfejsu ISP/JTAG. Opis pinów mikrokontrolera ATmega32A dla interfejsu JTAG [27]</p> <p>Poszczególne wyjścia programatora zostały podłączone do mikrokontrolera zgodnie z powyższym rysunkiem z 2.7. za pomocą interfejsu JTAG.</p> <p>Programator został podłączony do komputera poprzez Hub USB znajdujący się w zestawie uruchomieniowym.</p>	MPLAB Snap			DEBUG		ATmega32A	Connector	Pin #	Pin Name	AVR® JTAG	AVR ISP(&DW)	Pin # & Pin Name		1	TVPP				2	TVDD	VTG	VTG	VCC	3	GND	GND	GND	GND	4	PGD	TDO	MISO	PC4 (TDO)	5	PGC	TCK	SCK	PC2 (TCK)	6	TAUX	RESET	RESET	RESET	7	TTDI	TDI	MOSI	PC5 (TDI)	8	TTMS	TMS		PC3 (TMS)
MPLAB Snap			DEBUG		ATmega32A																																																	
Connector	Pin #	Pin Name	AVR® JTAG	AVR ISP(&DW)	Pin # & Pin Name																																																	
	1	TVPP																																																				
	2	TVDD	VTG	VTG	VCC																																																	
	3	GND	GND	GND	GND																																																	
	4	PGD	TDO	MISO	PC4 (TDO)																																																	
	5	PGC	TCK	SCK	PC2 (TCK)																																																	
	6	TAUX	RESET	RESET	RESET																																																	
	7	TTDI	TDI	MOSI	PC5 (TDI)																																																	
	8	TTMS	TMS		PC3 (TMS)																																																	

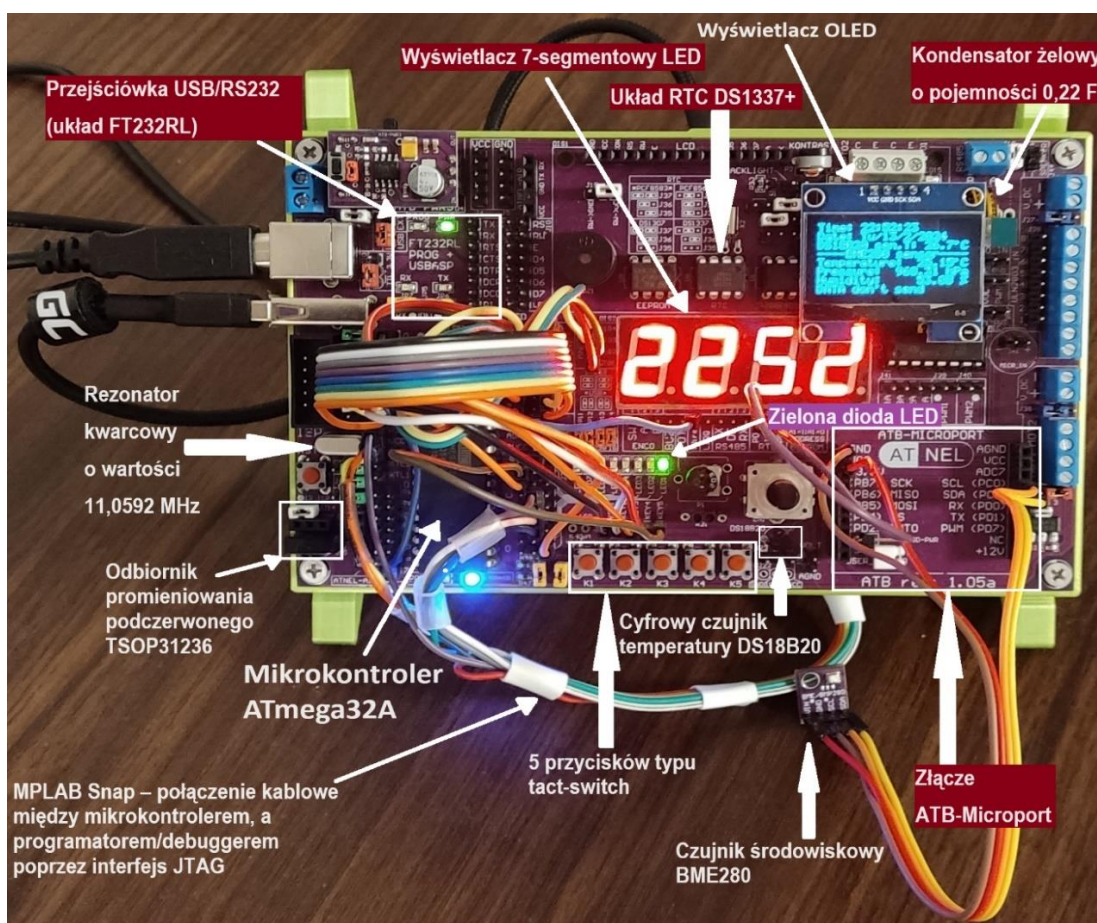


### 3. CHARAKTERYSTYKA STACJI MONITORUJĄCEJ PARAMETRY POWIETRZA

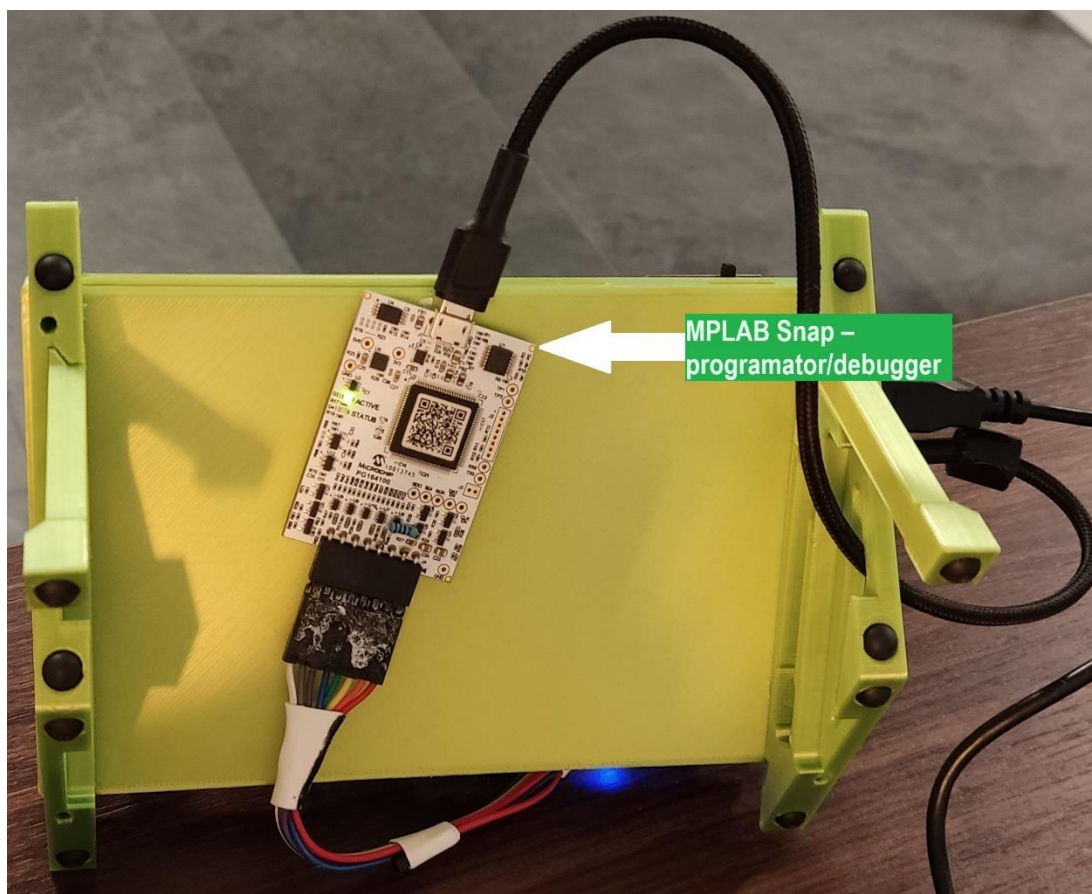
W tym rozdziale zaprezentowano gotowy model stacji monitorującej parametry powietrza wraz z podłączonymi oraz uruchomionymi modułami elektronicznymi. Przedstawiono również wszystkie funkcjonalności urządzenia wraz z ich szczegółowym opisem.

#### 3.1. Zaprezentowanie stacji pomiarowej stworzonej na bazie zestawu uruchomieniowego Atnel ATB 1.05a Andromeda

Przygotowane urządzenie oraz kod zostało stworzone ze szczególnym naciskiem na uniwersalność oraz niezawodność. Kod został zoptymalizowany pod kątem wykorzystanego mikrokontrolera oraz wykorzystanych modułów elektronicznych. Ze względu na to, że urządzenie ma mieć zastosowanie w analizie parametrów powietrza, dużą rolę odgrywała stabilność kodu. Jest to niezwykle ważne, gdyż późniejsze jego implementacje mogą zostać wykorzystywane w dziedzinach, w których precyzja pomiarów jest istotna, na przykład systemy zarządzania budynkiem. Na rysunku 3.1. oraz na rysunku 3.2. została przedstawiona stacja monitorująca parametry powietrza, która została zbudowana na bazie zestawu uruchomieniowego Atnel ATB 1.05a Andromeda. Zdjęcia te przedstawiają również moduły elektroniczne opisywane w poprzednich podrozdziałach.



Rys. 3.1. Stacja monitorująca parametry powietrza - opis elementów



Rys. 3.2. Stacja monitorująca parametry powietrza - programator/debugger MPLAB Snap

Na powyższych rysunkach 3.1. oraz 3.2. zostały wskazane następujące komponenty elektroniczne:

- rezonator kwarcowy o wartości 11,0592 MHz;
- kondensator żelowy o pojemności 0,22 F;
- układ RTC DS1337+;
- wyświetlacz OLED;
- wyświetlacz 7-segmentowy LED w kolorze czerwonym, posiadający cztery cyfry oraz cztery kropki;
- czujnik środowiskowy BME280;
- 5 przycisków typu tact-switch;
- przejściówka USB/RS232 (układ FT232RL);
- odbiornik promieniowania podczerwonego TSOP31236;
- zielona dioda LED;
- złącze ATB-Microport;
- MPLAB Snap – programator/debugger dla mikrokontrolerów Microchip.

Na powyższych rysunkach 3.1. oraz 3.2. urządzenie znajduje się w specjalnej obudowie od firmy Atmel, która ułatwia pracę z zestawem oraz pozytywnie wpływa na bezpieczeństwo elektroniki na dolnej stronie PCB. Obudowa ta chroni płytkę drukowaną przed naprężeniami mechanicznym, które mogą uszkodzić ścieżki drukowane, połączenia lutownicze oraz elementy elektroniczne.

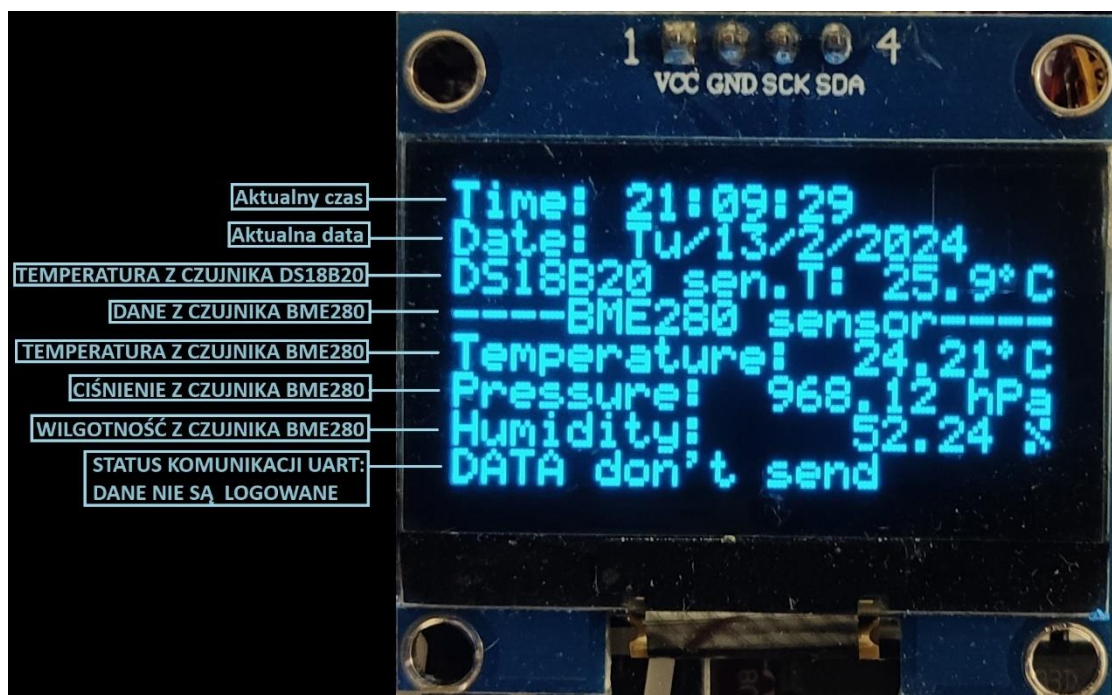
### 3.2. Omówienie funkcjonalności stacji pomiarowej

Na podstawie założeń projektu oraz gotowego modelu stacji został przygotowany opis techniczny omawianego urządzenia. Głównym zadaniem stacji jest monitorowanie oraz zbieranie w czasie rzeczywistym danych o parametrach powietrza, takich jak: temperatura, ciśnienie atmosferyczne oraz wilgotność. Sercem stacji pomiarowej jest mikrokontroler z rodziny AVR – ATmega32A. Bardzo ważnym modulem zewnętrznym jest zegar czasu rzeczywistego DS1337+, którego zadaniem jest synchronizowanie poszczególnych zadań.

Wyświetlacz OLED odpowiada za interfejs użytkownika. To właśnie na nim wyświetlane są najważniejsze informacje bez konieczności komunikowania się z urządzeniem poprzez interfejs UART. Na wyświetlaczu znajdziemy takie informacje jak:

- aktualna czas z zegara czasu rzeczywistego w formacie 24-godzinnym - HH:mm:ss (godziny:minuty:sekundy);
- aktualna data z zegara czasu rzeczywistego w formacie dzień/dd/MM/yyyy (dzień tygodnia/data/miesiąc/rok);
- aktualna wartość temperatury z cyfrowego czujnika temperatury DS18B20 w °C;
- aktualna wartość temperatury powietrza w °C, ciśnienia atmosferycznego w hPa oraz wilgotności powietrza z czujnika środowiskowego BME280 w %;
- stan wysyłania danych poprzez komunikację UART;
- informacja o kontroli urządzenia poprzez pilot podczerwieni.

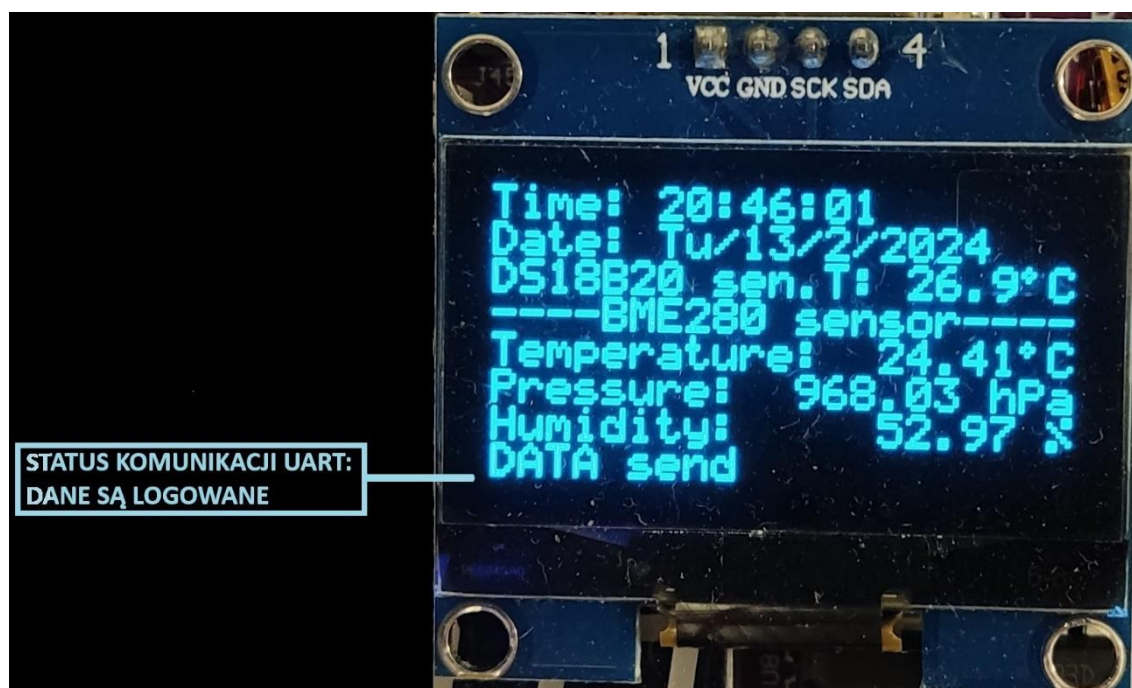
Na rysunku 3.3. został przedstawiony domyślny stan wyświetlacza, gdy dane nie są logowane poprzez komunikację UART. W takiej sytuacji na ekranie jest widoczna informacja: „DATA don't send”, czyli dane nie są logowane. Dane na wyświetlaczu są wyświetlane w języku angielskim oraz zostały opisane w języku polskim na rysunku poniżej.



Rys. 3.3. Domyślny stan wyświetlacza stacji pomiarowej

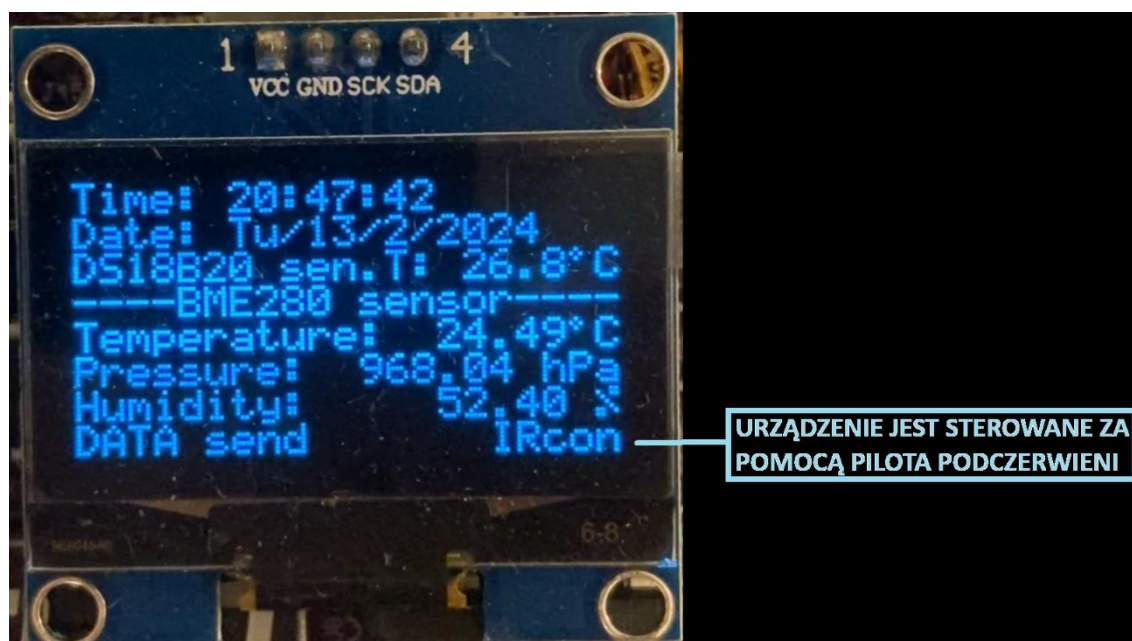


Na rysunku 3.4. został przedstawiony stan wyświetlacza, gdy dane są logowane poprzez komunikację UART. W takiej sytuacji na ekranie jest widoczna informacja: „DATA send”, czyli dane są logowane.



Rys. 3.4. Stan wyświetlacza stacji pomiarowej, gdy dane są logowane poprzez komunikację UART

Na rysunku 3.5 został przedstawiony stan wyświetlacza, gdy urządzenie jest sterowane za pomocą pilota podczerwieni. W takiej sytuacji na ekranie jest widoczna informacja: „IRcon”, czyli urządzenie jest kontrolowane z wykorzystaniem podczerwieni.



Rys. 3.5. Stan wyświetlacza stacji pomiarowej, gdy jest ona kontrolowana z wykorzystaniem podczerwieni

Na wyświetlaczu LCD wyświetlany jest aktualny czas z układu RTC. Cyfry numer jeden oraz dwa czerwonego wyświetlacza siedmiosegmentowego wyświetlają aktualną godzinę, natomiast cyfry numer trzy oraz cztery tego wyświetlacza wskazują minuty. Zastosowany został efekt pulsowania dla minut, który polega na tym, że cyfry odpowiedzialne za wyświetlanie minut są włączane oraz wyłączane w rytm sekund. Dzięki temu na wyświetlaczu 7-segmentowym dostępny jest również sekundnik.

Czujnik środowiskowy BME280 wykorzystuje interfejs komunikacyjny I<sup>2</sup>C oraz przekazuje wartości o parametrach takich jak temperatura powietrza, ciśnienie atmosferyczne oraz wilgotność powietrza z częstotliwością 1 Hz, czyli w każdej sekundzie wykonywany jest pomiar i odczyt powyższych parametrów. Użytkownik dane pomiarowe może obserwować za pomocą wyświetlacza OLED lub za pomocą terminala portu szeregowego z wykorzystaniem komunikacji UART.

Cyfrowy czujnik temperatury DS18B20 pracuje na magistrali 1-Wire oraz informuje o aktualnej wartości temperatury otoczenia. W jednej sekundzie mikrokontroler wysyła polecenie do czujnika o odczyt temperatury, natomiast w kolejnej sekundzie czujnik zwraca aktualną wartość temperatury, która jest wyświetlana użytkownikowi na wyświetlaczu OLED oraz może zostać przekazana za pomocą komunikacji UART.

Jako wskaźnik informujący o pracy urządzenia została zastosowana zielona migająca dioda LED z częstotliwością 2 Hz. Może ona zostać włączona, gdy wyłączymy wyświetlacz OLED oraz LED, lecz nadal chcemy mieć informację, że urządzenie działa prawidłowo.

Standardowym sposobem interakcji użytkownika oraz stacji pomiarowej są fizyczne przyciski. Została zaimplementowana obsługa pięciu przycisków, które wykonują dziesięć funkcji. Jest to możliwe dzięki pomiarze czasu naciśnięcia przycisków. W ten sposób urządzenie rozpoznaje krótkie oraz długie naciśnięcie wybranego klawisza rozpoznaje krótkie oraz długie naciśnięcie wybranego klawisza oraz wykonuje przypisaną mu funkcję.

Lista funkcji została opisana poniżej w tabeli 3.1.

**Tabela 3.1.** Funkcje realizowane za pomocą fizycznych przycisków

Numer przycisku	Realizowana funkcja
K1	Włączenie oraz wyłączenie wyświetlacza OLED.
K2	Włączenie oraz wyłączenie wyświetlacza 7-segmentowego LED.
K3	Włączenie oraz wyłączenie migającej, zielonej diody LED.
K4	Włączenie oraz wyłączenie logowanie danych poprzez komunikację UART.
K5	Jednoczesne włączenie lub wyłączenie: <ul style="list-style-type: none"> <li>wyświetlacza OLED;</li> <li>wyświetlacza 7-segmentowego LED;</li> <li>migającej, zielonej diody LED;</li> <li>logowania danych poprzez komunikację UART.</li> </ul>

Krótkie naciśnięcie przycisku odpowiada za włączenie danej funkcji, natomiast długie przyciśnięcie przycisku powoduje wyłączenie wybranej funkcji. Dzięki takiej implementacji komunikacja z urządzeniem jest przyjazna dla użytkownika.

Komunikacja ze stacją pomiarową została również zrealizowana dzięki implementacji odbiornika promieniowania podczerwonego. Został tutaj wykorzystany protokół RC5, który umożliwia sterowanie za pomocą pilota podczerwieni, czyli bezprzewodowo. Tak jak w przypadku fizycznych przycisków, użytkownik ma do dyspozycji dziesięć funkcji, które są przypisane do fizycznych klawiszy pilota podczerwieni. Mając na uwadze uniwersalność, funkcje zostały przypisane do klawiszy od numeru 0, do numeru 9 (klawisze te odpowiadają za wybór programów w standardowym odbiorniku telewizyjnym). Lista funkcji realizowanych za pomocą przycisków pilota podczerwieni została przedstawiona poniżej w tabeli 3.2.

**Tabela 3.2.** Funkcje realizowane za pomocą przycisków pilota podczerwieni

Numer przycisku pilota podczerwieni	Realizowana funkcja
0	Włączenie wyświetlacza OLED.
1	Wyłączenie wyświetlacza OLED.
2	Włączenie wyświetlacza 7-segmentowego LED.
3	Wyłączenie wyświetlacza 7-segmentowego LED.
4	Włączenie migającej, zielonej diody LED.
5	Wyłączenie migającej, zielonej diody LED.
6	Włączenie logowanie danych poprzez komunikację UART.
7	Wyłączenie logowanie danych poprzez komunikację UART.
8	Jednoczesne włączenie: <ul style="list-style-type: none"> <li>• wyświetlacza OLED;</li> <li>• wyświetlacza 7-segmentowego LED;</li> <li>• migającej, zielonej diody LED;</li> </ul> logowania danych poprzez komunikację UART.
9	Jednoczesne wyłączenie: <ul style="list-style-type: none"> <li>• wyświetlacza OLED;</li> <li>• wyświetlacza 7-segmentowego LED;</li> <li>• migającej, zielonej diody LED;</li> </ul> logowania danych poprzez komunikację UART.

Komunikacja zdalna z urządzeniem jest również jest możliwa, dzięki implementacji komunikacji UART o prędkości 19200 bitów na sekundę. Komunikacja ta działa dwustronnie, ponieważ umożliwia ona na sterowanie urządzeniem z informacją zwrotną o statusie poszczególnych funkcji. Dla wygody użytkownika, stacja pomiarowa został wyposażona w przyjazny system komunikacji zdalnej. Użytkownik może wysłać do urządzenia komendy do sterowania poszczególnymi funkcjami oraz może kontrolować urządzenie, tak jak w przypadku przycisków fizycznych oraz pilota podczerwieni. Dużą zaletą tej komunikacji jest to, że nie musimy fizycznie przebywać ze stacją, aby ją kontrolować. Lista dostępnych komend dla komunikacji

UART oraz opis realizowanych funkcji znajduje się poniżej w tabeli 3.3.

**Tabela 3.3.** Lista dostępnych komend dla komunikacji UART oraz opis realizowanych funkcji

Komenda dla komunikacji UART	Realizowana funkcja
„h” lub „H”	Wyświetlenie pomocy użytkownika
„0”	Włączenie wyświetlacza OLED.
„1”	Wyłączenie wyświetlacza OLED.
„2”	Włączenie wyświetlacza 7-segmentowego LED.
„3”	Wyłączenie wyświetlacza 7-segmentowego LED.
„4”	Włączenie migającej, zielonej diody LED.
„5”	Wyłączenie migającej, zielonej diody LED.
„6”	Włączenie logowanie danych poprzez komunikację UART.
„7”	Wyłączenie logowanie danych poprzez komunikację UART.
„8”	Jednoczesne włączenie: <ul style="list-style-type: none"> <li>• wyświetlacza OLED;</li> <li>• wyświetlacza 7-segmentowego LED;</li> <li>• migającej, zielonej diody LED;</li> </ul> logowania danych poprzez komunikację UART.
„9”	Jednoczesne wyłączenie: <ul style="list-style-type: none"> <li>• wyświetlacza OLED;</li> <li>• wyświetlacza 7-segmentowego LED;</li> <li>• migającej, zielonej diody LED;</li> </ul> logowania danych poprzez komunikację UART.

W przypadku gdy zostanie włączona funkcja logowania danych poprzez stację pomiarową, w terminalu portu szeregowego uzyskamy informację zwrotną z urządzenia o statusie poszczególnych funkcji oraz aktualne dane z zegara czasu rzeczywistego oraz z zewnętrznych czujników. Uzyskane w ten sposób dane pomiarowe są przystosowane do dalszego przetwarzania w programach zewnętrznych, a stacja umożliwia na rejestrowanie następujących danych:

- aktualna data oraz czas z zegara czasu rzeczywistego;
- aktualna wartość temperatury z cyfrowego czujnika temperatury DS18B20;
- aktualna wartość temperatury powietrza, ciśnienia atmosferycznego oraz wilgotności powietrza z czujnika środowiskowego BME280;
- informacja zwrotna o poszczególnych funkcjach, jeśli użytkownik w trakcie logowania danych korzystał z urządzenia.

Kontrolowanie urządzenia za pomocą przycisków fizycznych, pilota podczerwieni oraz komunikacji UART może być realizowane w tym czasie.

## 4. ANALIZA KODU ŹRÓDŁOWEGO STACJI POMIAROWEJ

W poniższym rozdziale omówiono biblioteki dla modułów elektronicznych wykorzystanych w projekcie. Zaprezentowano również zasadę działania poszczególnych fragmentów kodu źródłowego przygotowanego dla stacji monitorującej parametry powietrza.

### 4.1. Wstęp do kodu oraz omówienie bibliotek wykorzystanych w projekcie

Kod źródłowy został opracowany na podstawie książek, specyfikacji technicznych oraz zasobów internetowych. Został on skompilowany oraz zoptymalizowany dla mikrokontrolera ATmega32A. Ze względu na uniwersalność kodu, został on stworzony w języku angielskim. Składnia języka C opiera się właśnie na tym języku, a w związku z tym definicje preprocesora, nazwy zmiennych, nazwy funkcji jak i komentarze dla poszczególnych linii zostały w pełni zapisane w języku angielskim. Kod główny programu został również zoptymalizowany pod kątem wizualnym, aby był bardziej przejrzysty i łatwiejszy do analizy. Analizowany w kolejnym podrozdziale kod źródłowy to implementacja algorytmu „round robin”. Algorytm ten polega na karuzelowym sposobie szeregowania procesów w systemie operacyjnym oraz nadaje każdemu procesowi ściśle określone przedziały czasowe bez uwzględniania priorytetów. W związku z tym, wszystkie procesy mają ten sam priorytet. Dużą zaletą algorytmu *round robin* jest jego prostota i łatwość implementacji programowej. Stacja pomiarowa jest urządzeniem wielozadaniowym, ponieważ wykonuje kilka procesów w taki sposób, aby żaden z nich nie zakłócał pracy innego. Funkcjonalności urządzenia, które mogą być wykonywane w tym samym czasie zostały opisane w podrozdziale 3.2. *Omówienie funkcjonalności stacji pomiarowej.*

W bieżącym podrozdziale przedstawiono biblioteki, których funkcje zostały zastosowane do stworzenia kodu źródłowego, który zostanie omówiony w kolejnym podrozdziale. Kod źródłowy w pliku *main.c* posiada 902 linijki, a dodatkowo duża część kodu znajduje się w bibliotekach dla poszczególnych modułów. Biblioteki użyte w programie pochodzą z różnych źródeł i na potrzeby mojego urządzenia zostały odpowiednio zmodyfikowane. Poniżej krótko opisano każdą z bibliotek oraz przedstawiono autorów oryginalnych plików:

- *1Wire* – biblioteka została omówiona przez Mirosława Kardasia w autorskiej książce [31] dla komunikacji 1-Wire, przeznaczona dla czujnika temperatury DS18B20;
- *7\_LED* – biblioteka została stworzona oraz omówiona przez Mirosława Kardasia w autorskiej książce [31]. Jest przeznaczona do sterowania czterema wyświetlaczami siedmiosegmentowymi;
- *I2C\_TWI* - biblioteka została stworzona oraz omówiona przez Mirosława Kardasia w autorskiej książce [31]. Jest przeznaczona do obsługi magistrali dla I<sup>2</sup>C dla układu RTC oraz pamięci EEPROM;
- *IR\_DECODE* - biblioteka została stworzona oraz omówiona przez Mirosława Kardasia w autorskiej książce [31]. Jest przeznaczona do obsługi kodowania RC5 (komunikacja poprzez podczerwień);
- *LCD* - biblioteka została stworzona oraz omówiona przez Mirosława Kardasia w autorskiej książce [31]. Jest przeznaczona do obsługi wyświetlacza LCD opartego na



sterowniku HD44780. Nie jest ona kompilowana w projekcie, natomiast jest możliwość wyświetlania danych ze stacji pomiarowej na wyświetlaczu LCD;

- *MK\_PRESSURE\_HUMIDITY\_LIB* - biblioteka została stworzona przez Mirosława Kardasia. Została ona zakupiona z oficjalnej strony firmy Atmel [32, 33]. Jest ona przeznaczona do obsługi czujników firmy Bosch takich jak: BMP085, BMP180, BMP280 oraz BME280 oraz czujnika firmy Honeywell, takiego jak HIH6131. Biblioteka została zaimplementowana, aby odczytywać temperaturę powietrza, ciśnienie atmosferyczne oraz wilgotność powietrza z czujnika BME280. Funkcje biblioteki zostały zmodyfikowane, aby poprawnie odczytywać ujemne temperatury;
- *MKUART* - biblioteka została stworzona oraz omówiona przez Mirosława Kardasia w autorskiej książce [31]. Jest przeznaczona do obsługi komunikacji UART;
- *OLED* – biblioteka została znaleziona w zasobach internetu [34]. Jej zadaniem jest obsługa wyświetlacza OLED poprzez magistralę I<sup>2</sup>C. Została dostosowana do moich potrzeb (poprawne wyświetlanie dużych liczb dla ciśnienia atmosferycznego).

Poza bibliotekami, w oknie przeglądu projektu, widoczne są jeszcze takie foldery, jak: *Debug* oraz *Release* zawierające gotowe pliki wynikowe po kompilacji programu. Są tutaj również widoczne pliki, takie jak: *common.h* (zawierający makra upraszczające dostęp do portów wyświetlacza LED – obecnie nieużywany), *main.c* (zawiera kod źródłowy programu) oraz *Praca dyplomowa.atsln*, *Praca dyplomowa.cproj* (można za ich pomocą uruchomić gotowy projekt).

#### **4.2. Omówienie kodu źródłowego projektu**

Ze względu na rozbudowany projekt, wszystkie procesy zostaną przedstawione w mniejszych partiach kodu źródłowego, aby jego analiza była bardziej przejrzysta oraz intuicyjna. Dodatkowo kod źródłowy został dokładnie opisany odpowiednimi komentarzami bezpośrednio w projekcie. Ze względu na czytelność, część kodów została wklejona bez wszystkich komentarzy, które są dostępne w pliku źródłowym *main.c*.

W liniach numer 9 – 15 zostały dołączone do programu standardowe biblioteki dla języka C, przeznaczone między innymi dla systemów wbudowanych. W liniach numer 19 – 28 zostały dołączone pliki nagłówkowe dla poszczególnych bibliotek opisanych powyżej. Każdy z tych plików nagłówkowych zawiera deklaracje funkcji, które mogą zostać użyte w programie głównym. Omawiany fragment kodu znajduje się na rysunku 4.1.

```

8  /*****INCLUSION OF STANDARD LIBRARIES*****/
9  #include <avr/io.h>
10 #include <avr/interrupt.h>
11 #include <avr/pgmspace.h>
12 #include <util/delay.h>
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <string.h>
16
17
18 /*****INCLUSION OF CUSTOM LIBRARIES*****/
19 #include "1Wire/ds18x20.h" // library for DS18B20 (1-Wire)
20 #include "7_LED/7_led.h" // library for LED display
21 #include "I2C_TWI/i2c_twi.h" // library for RTC (I2C)
22 #include "IR_DECODE/ir_decode.h" // library for IR pilot (RC5)
23 // #include "LCD/lcd44780.h" // switched off from compilation - LCD display is not used in this project
24 #include "MK_PRESSURE_HUMIDITY_LIB/mk_i2c.h" // library for BME280 (I2C)
25 #include "MK_PRESSURE_HUMIDITY_LIB/mk_pressure_cfg.h" // library for BME280 (reading parameters via I2C)
26 #include "MKUART/mkuart.h" // library for UART communication
27 #include "OLED/lcd.h" // library for OLED display
28 #include "OLED/font.h" // library for OLED display

```

Rys. 4.1. Dołączenie bibliotek standardowych oraz plików nagłówkowych

W dalszej części kodu przedstawionej na rysunku 4.2. znajdują się definicje preprocesora (linie numer 32 – 53). W definicjach preprocesora znajduje się adres układu RTC na magistrali I<sup>2</sup>C (0xD0), przypisanie portu dla migającej zielonej diody LED oraz przypisanie przycisków fizycznych do poszczególnych pinów mikrokontrolera.

```

31 /*****PREPROCESSORS DEFINITIONS*****/
32 #define DS1337_ADDR 0xD0 // address of the RTC chip on I2C
33
34 #define LED1 (1<<PC6) // standard LED definition
35
36 #define LED1_DDR DDRC
37
38 #define LED1_ON PORTC &= ~LED1
39 #define LED1_OFF PORTC |= LED1
40 #define LED1_TOG PORTC ^= LED1
41
42
43 #define KL_DDRA DDRA // buttons definition
44 #define KL_DDRC DDRC
45
46 #define KL_PORTA PORTA
47 #define KL_PORTC PORTC
48
49 #define KEY1 (1<<PA4)
50 #define KEY2 (1<<PA5)
51 #define KEY3 (1<<PA6)
52 #define KEY4 (1<<PA7)
53 #define KEY5 (1<<PC7)

```

Rys. 4.2. Fragment kodu źródłowego zawierający definicje preprocesora

Rysunek 4.3. przedstawia definicje typów oraz zmiennych globalnych (linie numer 57 – 88). Część zmiennych globalnych, które są używane w przerwaniach, mają dodany specyfikator *static* oraz *volatile*. Przydomek *static* (ang. „statyczny”) oznacza zmienną statyczną.

Wykorzystanie go przy zmiennych globalnych spowoduje, że użyte zmienne przestaną być widoczne w innych plikach. Natomiast przydomek *volatile* (ang. „ulotny”) sprawia, że kompilator nie optymalizuje dostępu do takich zmiennych. Gdy kompilator chce wykonać na nich operację, to musi się odwołać bezpośrednio do zawartości komórek pamięci, w których te zmienne się znajdują. Następnie w kolejnym kroku ich zawartość może zostać zmodyfikowana. W związku z tym, takie zmienne mogą być zapisywane i odczytywane w funkcjach oraz w procedurach przerwań mikrokontrolera. Zmienna ze specyfikatorem *volatile* może się zmieniać w różnym czasie. Ze względu na ilość zmiennych, ich szczegółowy opis został zawarty w kodzie źródłowym.

```

56  /*****DEFINITIONS OF GLOBAL VARIABLES AND TYPES*****/
57  uint8_t key1, key2, key3, key4, key5;    // variables for SuperDebounce function
58
59  volatile static uint8_t led_display = 1;  // variable responsible for switching on/off LED display
60  volatile static uint8_t led_blink; // variable responsible for switching on/off LED blinking
61
62  volatile static uint8_t command_cnt;    // variable responsible for counting cycles of IR receiver events
63
64  volatile static char uart_tx; // data from UART (RX data)
65  volatile static uint8_t uart_help = 1; // variable responsible for UART help
66  volatile static uint8_t uart_logs_data; // variable responsible for UART logging data
67
68  volatile static uint16_t Timer1, Timer2, Timer3, Timer4; // software timers 100Hz
69
70  uint8_t RTC_modify = 0; // RTC_modify == 0 -> don't modify time and date with every flashing / reset of device,
71  enum {SS, MIN, HH, DAY, DD, MM, YY}; // enumerated type for seconds, minutes, hours, day of the week, and date
72  uint8_t buffer[7]; // table/buffer (saving data to RTC chip)
73  uint8_t ss, min, hh, day, dd, mm, yy; // variables to read data from RTC chip
74  static volatile uint8_t ss_memory; // variable to save difference for time
75
76  volatile static uint8_t int0_flag=1; // flag changed in the interrupt and checked in the main loop (RTC event)
77
78  uint8_t subzero, cel, cel_fract_bits; // variables for DS18B20
79  uint8_t sensors_cnt; // quantity of sensors on the 1-Wire bus
80  volatile static uint8_t cycle; // variable to start measuring and read the temperature from DS18B20
81
82  int8_t t_int, t_fract; // variables to read and display the value of temperature
83  volatile static uint8_t t_fract_flag = 0; // information about minus temperature after the decimal point for UART
84
85  uint16_t hp_int; // variables to read and display the value of pressure
86  uint8_t hp_fract;
87
88  uint8_t hm_int, hm_fract; // variables to read and display the value of humidity

```

Rys. 4.3. Definicje typów oraz zmiennych globalnych

Na rysunku 4.4. znajduje się fragment kodu z deklaracją funkcji oraz procedur, które zostały zapisane w liniach numer 92 – 122. Są to w większości funkcje oraz procedury przygotowane na potrzeby tego projektu. Funkcje tutaj przywołane, odpowiadają za obsługę przycisków fizycznych, wyświetlanie danych na wyświetlaczu LED/OLED, obsługę komunikacji UART oraz za obsługę komunikacji poprzez podczerwień. Procedury korzystają również z funkcji bibliotecznych (z bibliotek dołączonych do projektu). Definicje funkcji oraz procedur znajdują się na samym końcu pliku głównego main.c (linie numer 432 – 902). Można tam przeanalizować poszczególne funkcje oraz procedury.

```

91  /*****FUNCTIONS DECLARATION*****/
92  void SuperDebounce(uint8_t * key_state, volatile uint8_t *KPIN, uint8_t key_mask, uint16_t rep_time,
93  uint16_t rep_wait, void (*push_proc)(void), void (*rep_proc)(void) );
94
95  void oled_display_temp_DS18B20(uint8_t y); // displaying a temperature from DS18B20 sensor on OLED display
96
97  void oled_day_of_the_week(uint8_t day); // convert a number of day to day of the week for OLED display
98  void uart_day_of_the_week(uint8_t day); // convert a number of day to day of the week for UART
99
100 void uart_control(void); // function for receiving data from UART and device control
101
102 void ir_control(void); // function for receiving data from RC5 and device control
103
104 void uart_data_logging(void); // function for sending measurements from sensors via UART
105
106 void key1_press(void); // function assigned to a button 1 (short press)
107 void key1_repeat(void); // function assigned to a button 1 (long press)
108
109 void key2_press(void); // function assigned to a button 2 (short press)
110 void key2_repeat(void); // function assigned to a button 2 (long press)
111
112 void key3_press(void); // function assigned to a button 3 (short press)
113 void key3_repeat(void); // function assigned to a button 2 (long press)
114
115 void key4_press(void); // function assigned to a button 4 (short press)
116 void key4_repeat(void); // function assigned to a button 4 (long press)
117
118 void key5_press(void); // function assigned to a button 5 (short press)
119 void key5_repeat(void); // function assigned to a button 5 (long press)
120
121 uint8_t dec2bcd(uint8_t dec); // convert decimal to BCD
122 uint8_t bcd2dec(uint8_t bcd); // convert BCD to decimal

```

Rys. 4.4. Definicje funkcji oraz procedur

Na rysunku 4.5. zaprezentowano początek głównej funkcji programu. Przedstawiona została tutaj inicjalizacja kierunku pinów w liniach numer 130 - 138 (wejścia dla przycisków fizycznych oraz wyjścia dla zielonej, migającej diody) oraz inicjalizacja przerwań dla mikrokontrolera ATmega32A. Przerwanie jest zdarzeniem, które przerywa wykonywanie programu głównego i uruchamia specjalną funkcję obsługi przerwania. Gdy funkcja obsługuje przerwanie, następuje powrót do przerwanego programu głównego i następuje wznowienie jego wykonywania od miejsca, w którym został przerwany [34]. Przerwanie *INT0* (linie numer 144 – 146) obsługuje pin mikrokontrolera *PD2*. Jest to przerwanie, które jest generowane zewnątrz przez układ RTC dokładnie co jedną sekundę. Przerwanie jest generowane zboczem opadającym [39]. Jest to bardzo ważne przerwanie w opracowanym w ramach pracy projekcie, ponieważ precyzyjne ustawianie podstawy czasu co jedną sekundę jest kluczowe dla działania innych układów oraz procesów w projekcie. Procedura obsługi tego przerwania znajduje się w liniach numer 406 – 409 oraz została przedstawiona na rysunku 4.6. Polega ona na ustawieniu flagi *int0\_flag* na wartość „1”. Flaga ta jest zerowana pod koniec każdego zdarzenia z układu RTC – zostało to zaprezentowane na rysunku 4.7. w 393. linii kodu źródłowego. W kodzie głównym programu również został zainicjalizowany sprzętowy Timer2 (linie numer 149 – 152), który jest 8-bitowy. Wykonuje on przerwanie co 10 ms (100 Hz) w trybie CTC. Procedura obsługi przerwania znajduje się w liniach numer 413 – 428 oraz na rysunku 4.8. Zostały tam stworzone cztery 16-bitowe Timery programowe, które zostały wykorzystane między innymi do obsługi przycisków fizycznych oraz zielonej, migającej diody. Na tym etapie programu warto wspomnieć, że sprzętowy Timer0 (8-bitowy) został wykorzystany do obsługi czterech wyświetlaczy 7-segmentowych, natomiast sprzętowy Timer1 (16-bitowy) został wykorzystany do obsługi

odbiornika promieniowania podczerwonego. Inicjalizację tych timerów sprzętowych można znaleźć w odpowiednich kodach źródłowych dla wyświetlacza siedmiosegmentowego (*7\_led.c*) oraz odbiornika podczerwieni (*ir\_decode.c*).

```

125  /*****MAIN FUNCTION OF PROGRAM*****/
126  int main(void)
127  {
128
129  /*-----INITIALIZATION OF PINS DIRECTION-----*/
130      KL_DDRA &= ~(KEY1|KEY2|KEY3|KEY4); // the direction register of buttons as an input
131      KL_DDRC &= ~(KEY5);
132
133      KL_PORTA |= KEY1|KEY2|KEY3|KEY4; // switching on a pull-up to VCC
134      KL_PORTC |= KEY5; // switching on a pull-up to VCC
135
136
137      LED1_DDR |= LED1; // the direction register of LED as an output
138      LED1_OFF; // switch off LED
139
140
141  /*-----INTERRUPT INITIALIZATION-----*/
142
143      /* INT0 interrupt */
144      MCUCR |= (1<<ISC01); // falling edge triggering
145      GICR |= (1<<INT0); // interrupt unlock
146      PORTD |= (1<<PD2); // pulling the INT0 pin to VCC
147
148      /* Timer2 - interrupt initialization - 10 ms (100Hz) */
149      TCCR2 |= (1<<WGM21); // CTC work mode
150      TCCR2 |= (1<<CS22)|(1<<CS21)|(1<<CS20); // prescaler = 1024
151      OCR2 = 108; // interrupt comparison every 10ms (100Hz)
152      TIMSK = (1<<OCIE2); // interrupt unlock CompareMatch
153

```

Rys. 4.5. Inicjalizacja kierunku pinów oraz inicjalizacja przerwań dla mikrokontrolera ATmega32A

```

405  /*****INT0 interrupt service routine*****/
406  ISR( INT0_vect )
407  {
408      int0_flag = 1;
409  }

```

Rys. 4.6. Procedura obsługi przerwania INT0

```

393  |                                     int0_flag = 0; // reset a flag for RTC event

```

Rys. 4.7. Zerowanie flagi int0\_flag pod koniec zdarzenia z układu RTC

```

412  /*****Timer2 interrupt service routine*****/
413  ISR(TIMER2_COMP_vect)
414  {
415      uint16_t x;
416
417      x = Timer1;    /* 100Hz Timer1 */
418      if (x) Timer1 = --x;
419
420      x = Timer2;    /* 100Hz Timer2 */
421      if (x) Timer2 = --x;
422
423      x = Timer3;    /* 100Hz Timer3 */
424      if (x) Timer3 = --x;
425
426      x = Timer4;    /* 100Hz Timer4 */
427      if (x) Timer4 = --x;
428  }

```

Rys. 4.8. Procedura obsługi przerwania timera sprzętowego Timer2

Fragment kodu na rysunku 4.9. przedstawia inicjalizację modułów programowych (linie numer 156 – 171), takich jak: czujnik DS18B20, magistrala I<sup>2</sup>C, czujnik BME280, wyświetlacz OLED, wyświetlacz 7-segmentowy, odbiornik podczerwieni oraz komunikacja UART. Po globalnym odblokowaniu przerwań *sei()* w linii numer 173, można ustawić aktualny czas oraz datę, a następnie zapisać te informacje do pamięci RAM układu RTC (linie numer 176 – 187). Zapis aktualnej godziny oraz daty jest bardzo łatwy, ze względu na wykorzystanie typu wyliczeniowego *enum*. Za pomocą zmiennej *RTC\_modify* możemy określić, czy chcemy nadpisywać czas oraz datę po każdym resecie urządzenia lub po wgraniu programu do mikrokontrolera. Ze względu na obecność kondensatora podtrzymującego zasilanie dla układu RTC, czas został ustawiony jednorazowo bez konieczności jego nadpisywania.

```

155  /*****INITIALIZATION OF PROGRAM MODULES*****/
156  sensors_cnt = search_sensors(); // check how many DS18xxx is present on 1-Wire bus
157  DS18X20_start_meas( DS18X20_POWER_EXTERN, NULL ); // start measuring of temperature for DS18B20
158
159  i2cSetBitrate(100); // set the speed for I2C (kHz)
160
161  mk_press_hum_init(); // BME280 sensor initialization
162
163  lcd_oled_init(LCD_DISP_ON); // initialization of OLED and turn on
164  lcd_set_contrast(100); // set a contrast for OLED display
165  lcd_charMode(1); // set a font size
166
167  d_led_init(); // LED multiplexing initialization
168
169  ir_init(); // initialization of the RC5 infrared receiver
170
171  USART_Init(__UBRR); // UART initialization
172
173  sei(); // global interrupt unlock
174
175  /*****SEND AND SET TIME, DATA FOR RTC CHIP*****/
176  if (RTC_modify == 1) // RTC_modify == 0 -> don't modify time and date with every flashing / reset of device,
177  { // RTC_modify == 1 -> - modify time and date with every flashing / reset of device
178      buffer[HH] = dec2bcd(22); // hours
179      buffer[MIN] = dec2bcd(44); // minutes
180      buffer[SS] = dec2bcd(00); // seconds
181      buffer[DAY] = dec2bcd(1); // day of the week (1-7)
182      buffer[DD] = dec2bcd(19); // day
183      buffer[MM] = dec2bcd(2); // month
184      buffer[YY] = dec2bcd(24); // year
185
186      TWI_write_buf(DS1337_ADDR, 0x00, 7, buffer); // writing data to RAM of RTC chip (7 bytes from buffer
187  } // to RAM from address 0x00)

```

Rys. 4.9. Inicjalizacja modułów programowych, globalne odblokowanie przerwań, ustawienie czasu oraz daty

Kolejnym krokiem w analizie kodu źródłowego jest przejście do głównej, nieskończonej pętli programu – jej fragment znajduje się na rysunku 4.10. Na tym rysunku zostało przedstawione zastosowanie funkcji *SuperDebounce* stworzonej przez Mirosława Kardasia. Została ona zaimplementowana dla pięciu przycisków fizycznych (linie numer 197 – 201). Klawisze te realizują łącznie dziesięć funkcji ze względu na uwzględnienie czasu naciśnięcia przycisków [34]. Mikrokontroler w przerwaniu sprawdza czas naciśnięcia przycisku i realizuje przypisaną mu funkcję. Funkcja *SuperDebounce* wykorzystuje timer programowy *Timer1*, który został stworzony na timerze sprzętowym *Timer2*. W linii numer 206 została zastosowana procedura, której zadaniem jest interakcja z użytkownikiem poprzez komunikację UART. Dzięki niej użytkownik może skorzystać z pomocy dotyczącej obsługi urządzenia oraz kontrolować stację pomiarową za pomocą zdefiniowanych komend. W linii numer 211 znajduje się procedura, odpowiadająca za sterowanie urządzeniem poprzez podczerwień za pomocą określonych przycisków na pilocie podczerwieni. W liniach numer 216 – 220 znajduje się kod, który odpowiada za miganie zieloną diodą kontrolną stacji pomiarowej. Można ją włączyć, gdy zostały wyłączone wyświetlacze OLED oraz LED, ale chcemy mieć pewność, że urządzenia nadal pracuje poprawnie. Migającą diodę możemy załączyć za pomocą przycisku, pilota podczerwieni oraz komunikacji UART. Do obsługi diody wykorzystano timer programowy *Timer2*, który został stworzony na timerze sprzętowym *Timer2* (procedura obsługi przerwania timera sprzętowego *Timer2* znajduje się w liniach 413 – 428 oraz na rysunku 4.8.). Gdy wartość zmiennej *Timer2* = 0 (linia 216), to jej wartość jest równa 25, a stan diody jest zmieniany na przeciwny (jeśli funkcja została włączona – zmienna *led\_blink* = 1). Timer programowy *Timer2* co 10 milisekund dekrementuje wartość *Timer2* = 25. Wynika z tego, że jeden cykl odliczania dla *Timer2* wynosi 250 ms ( $25 \cdot 10$  ms). Stan diody zmienia się więc cztery razy na sekundę (2 razy dioda jest włączona, dwa razy dioda jest wyłączona), więc dioda miga z częstotliwością 2 Hz.

```

192  /*-----MAIN LOOP-----
193      while(1)
194      {
195
196          /******FUNCTIONALITY OF BUTTONS*****
197          SuperDebounce(&key1, &PINA, KEY1, 2, 15, key1_press, key1_repeat );
198          SuperDebounce(&key2, &PINA, KEY2, 2, 15, key2_press, key2_repeat );
199          SuperDebounce(&key3, &PINA, KEY3, 2, 15, key3_press, key3_repeat );
200          SuperDebounce(&key4, &PINA, KEY4, 2, 15, key4_press, key4_repeat );
201          SuperDebounce(&key5, &PINC, KEY5, 2, 15, key5_press, key5_repeat );
202
203
204
205          /******FUNCTIONALITY OF UART*****
206          uart_control(); // receiving data from UART and device control
207
208
209
210          /******INFRARED RECEIVER EVENT*****
211          ir_control(); // receiving data from RC5 and device control
212
213
214
215          /******FUNCTIONALITY OF SOFTWARE TIMER2*****
216          if(!Timer2) // task assigned for software Timer2
217          {
218              Timer2=25;
219              if(led_blink) LED1_TOG;
220          }

```

Rys. 4.10. Funkcja SuperDebounce do obsługi przycisków. Procedura do interakcji z użytkownikiem poprzez komunikację UART oraz procedura odpowiadająca za sterowanie urządzeniem poprzez podczerwień. Kod odpowiadający za miganie diodą kontrolną

Na rysunku 4.11. został przedstawiony fragment kodu, który jest realizowany w każdej sekundzie. Tak jak opisano wcześniej, układ RTC generuje przerwanie co jedną sekundę i ustawia flagę *int0\_flag* na wartość równą „1”. Na poniższym rysunku, w liniach numer 229 – 236 jest odczytywany aktualny czas oraz data z pamięci RAM układu RTC. Następnie w liniach numer 241 – 262 została napisana obsługa wyświetlacza 7-segmentowego LED, który wyświetla godziny oraz minuty (dwie cyfry minut migają w rytm sekund, co generuje efekt „pulsowania”). Efekt ten został uzyskany dzięki wyłączeniu cyfr numer trzy oraz cztery wyświetlacza 7-segmentowego dla sekund nieparzystych. Cyfry numer trzy oraz cztery wyświetlacza 7-segmentowego są włączane dla sekund parzystych. Włączanie oraz wyłączanie wyświetlacza LED jest możliwe dzięki zmiennej *led\_display*. Wartość „0” oznacza, że wyświetlacz jest wyłączony, natomiast wartość „1” włącza ponownie wyświetlacz.



```

224 /*-----RTC SYSTEM EVENT-----*/
225 if (int0_flag)
226 {
227
228     /*^^^^^^^^^^^^^^^^READ AND SET TIME, DATA FOR RTC CHIP^^^^^^^^^^^^^^^^*/
229     TWI_read_buf( DS1337_ADDR, 0x00, 7, buffer);    // reading data from RAM of RTC chip
230     hh = bcd2dec( buffer[HH] );                    //(7 bytes from RAM from address 0x00 to buffer)
231     min = bcd2dec( buffer[MIN] );
232     ss = bcd2dec( buffer[SS] );
233     day = bcd2dec( buffer[DAY] );
234     dd = bcd2dec( buffer[DD] );
235     mm = bcd2dec( buffer[MM] );
236     yy = bcd2dec( buffer[YY] );
237
238
239     /*^^^^^^^^^^^^^^^^FUNCTIONALITY OF CLOCK ON LED DISPLAY^^^^^^^^^^^^^^^^*/
240     if (led_display == 1) // show a clock on LED display
241     {
242         if (ss %2 != 0)    // if seconds %2 != 0, seconds on LED display are off
243         {
244             {
245                 cy1 = buffer[HH]>>4;    // displaying a time on OLED display (only hours)
246                 cy2 = buffer[HH]&0x0f;
247                 cy3=cy4=10;    // display nothing
248             }
249
250             else if (ss %2 == 0) // if seconds %2 == 0, seconds on LED display are on
251             {
252                 cy1 = buffer[HH]>>4;    // displaying a time on OLED display (hours and minutes)
253                 cy2 = buffer[HH]&0x0f;
254                 cy3 = buffer[MIN]>>4;
255                 cy4 = buffer[MIN]&0x0f;
256             }
257         }
258
259         if (led_display == 0) // doesn't show a clock on LED display
260         {
261             cy1=cy2=cy3=cy4=10;    // display nothing
262         }
263     }

```

Rys. 4.11. Odczytywanie danych z pamięci RAM układu RTC oraz obsługa wyświetlacza 7-segmentowego

Kolejny fragment kodu przedstawiony na rysunku 4.12. jest również realizowany w przerwaniu generowanym przez układ RTC. Ten fragment kodu odpowiada za wyświetlanie czasu oraz daty na wyświetlaczu OLED, który wykorzystuje interfejs komunikacyjny I<sup>2</sup>C oraz posiada adres 0x78. Obsługa wyświetlacza OLED jest prosta, gdyż zostały tutaj wykorzystane tylko trzy funkcje biblioteczne, takie jak:

- void lcd\_gotoxy(uint8\_t x, uint8\_t y); // ustaw kursor na pozycjach x, y;
- void lcd\_puts\_p(const char\* progmem\_s); // wyświetl string z pamięci FLASH (w trybie tekstowym) na wyświetlaczu OLED;
- void oled\_int(int val); // wyświetlenie wartości dziesiętnej na wyświetlaczu OLED.

Kod został przetestowany dla różnych konfiguracji czasu oraz daty.

```

266      /******FUNCTIONALITY OF OLED DISPLAY - TIME******/
267      lcd_gotoxy(0,0);    // displaying a time on OLED display
268      lcd_puts_p(PSTR("Time: "));
269      if( hh < 10 ) lcd_puts_p(PSTR("0"));
270      oled_int(hh);
271      lcd_puts_p(PSTR(":"));
272      if( min < 10 ) lcd_puts_p(PSTR("0"));
273      oled_int(min);
274      lcd_puts_p(PSTR(":"));
275      if( ss < 10 ) lcd_puts_p(PSTR("0"));
276      oled_int(ss);
277
278
279
280      /******FUNCTIONALITY OF OLED DISPLAY - DATE******/
281      lcd_gotoxy(0,1);    // displaying a date on OLED display
282      lcd_puts_p(PSTR("Date: "));
283      oled_day_of_the_week(day);
284      lcd_puts_p(PSTR("/"));
285      oled_int(dd);
286      lcd_puts_p(PSTR("/"));
287      oled_int(mm);
288      lcd_puts_p(PSTR("/"));
289      if (yy != 0) lcd_puts_p(PSTR("20"));
290      oled_int(yy);
291      if(dd < 10 || mm < 10) lcd_puts_p(PSTR(" "));

```

Rys. 4.12. Fragment kodu odpowiedzialny za wyświetlanie czasu oraz daty na wyświetlaczu OLED

Na rysunku 4.13. został przedstawiony fragment kodu obsługujący czujnik temperatury DS18B20, który wykorzystuje interfejs komunikacyjny 1-Wire. Obsługa czujnika polega na wysłaniu polecenia dokonania pomiaru przez czujnik. Polecenie to zostało wysłane podczas inicjalizacji modułów programowych, co zostało przedstawione na rysunku 4.9. w 157. linii kodu źródłowego. Następnie podczas zdarzenia z układu RTC, w 298. linii kodu źródłowego został dokonany odczyt temperatury. Tuż po odczycie tej wartości za pomocą interfejsu komunikacyjnego 1-Wire, ponownie wysłano polecenie dokonania pomiaru przez czujnik (linia numer 300). Dzięki takiej sekwencji poleceń, czas między wysłanym poleceniem pomiaru temperatury, a jej odczytem z czujnika jest dłuższy niż 750 ms. Pomiar jest dodatkowo wykonywany w każdej sekundzie. Następnie wartość temperatury można przedstawić na wyświetlaczu OLED za pomocą procedury `void oled_display_temp_DS18B20(uint8_t y)` - procedura ta została przedstawiona na rysunku 4.14. Została ona przygotowana do wykorzystanego w projekcie wyświetlacza OLED o rozdzielczości 128 × 64 pikseli. Jest do niej przekazywana tylko jedna zmienna – numer linii wyświetlacza, w której ma zostać wyświetlona temperatura z cyfrowego czujnika temperatury DS18B20. Wyświetlenie temperatury jest możliwe dzięki wcześniej wykorzystanej funkcji do odczytu temperatury `uint8_t DS18X20_read_meas(uint8_t *id, uint8_t *subzero, uint8_t *cel, uint8_t *cel_frac_bits)`. Została ona zaprezentowana w 298 linii kodu źródłowego na rysunku 4.13. Są do niej przekazywane wskaźniki do trzech jednobajtowych zmiennych typu `uint8_t`, które kolejno oznaczają:

- `subzero` – znak (temperatura dodatnia == 0 lub temperatura ujemna == 1);
- `cel` – wartość dziesiętna temperatury;
- `cel_frac_bits` – wartość temperatury po przecinku.

Procedura do wyświetlania temperatury na wyświetlaczu OLED wykorzystuje wartości zmiennych *subzero*, *cel* oraz *cel\_fract\_bits*. Końcowym efektem napisanej procedury jest prawidłowe wyświetlenie temperatury w zależności od wartości znaku, części dziesiętnej oraz ułamkowej temperatury. Omawiana procedura została sprawdzona z dynamicznie zmieniającymi się wartościami temperatury, aby uniknąć błędów związanych z jej wyświetlaniem.

```

295 | //*****FUNCTIONALITY OF DS18B20 SENSOR*****//
296 |
297 | // read the temperature from the sensor every second, if detected. Display the temperature when the sensor is detected
298 | if( DS18X20_OK == DS18X20_read_meas(gSensorIDs[0], &subzero, &cel, &cel_fract_bits) ) oled_display_temp_DS18B20(2);
299 |
300 | DS18X20_start_meas( DS18X20_POWER_EXTERN, NULL ); // start measuring of temperature for DS18B20

```

Rys. 4.13. Fragment kodu odpowiedzialny za obsługę czujnika temperatury DS18B20

```

493 | //*****Displaying a temperature from DS18B20 sensor on OLED display*****//
494 | void oled_display_temp_DS18B20(uint8_t y)
495 | {
496 |     lcd_gotoxy(0, y); // set the line for displaying
497 |     lcd_puts_p(PSTR("DS18B20 sen.T: ")); // display a name of sensor
498 |
499 |     if(subzero && cel < 10) lcd_gotoxy(14, y), lcd_puts_p(PSTR(" -")); // if subzero = 1 and cel < 10, display correctly sign "-"
500 |     else if (subzero && cel >= 10) lcd_gotoxy(14, y), lcd_puts_p(PSTR("-")); // if subzero = 1 and cel >= 10, display correctly sign "-"
501 |
502 |     if(!subzero && cel < 10) lcd_gotoxy(14, y), lcd_puts_p(PSTR(" ")); // set the cursor correctly when the temperature is smaller or greater than 10 or equal to 10
503 |     else if (!subzero && cel >= 10) lcd_gotoxy(14, y), lcd_puts_p(PSTR(" "));
504 |
505 |     oled_int(cel); // display decimals of temperature
506 |     lcd_puts_p(PSTR(".")); // display a dot
507 |     oled_int(cel_fract_bits); // display the number after the decimal point
508 |     lcd_puts_p(PSTR("C")); // display unit sign (C - degrees Celsius)
509 | }

```

Rys. 4.14. Procedura, której zadaniem jest wyświetlanie temperatury z czujnika DS18B20 na wyświetlaczu OLED

Obsługa czujnika środowiskowego BME280 została przedstawiona na rysunkach 4.15. oraz 4.16. Czujnik o adresie 0xEC za pomocą magistrali I<sup>2</sup>C komunikuje się z mikrokontrolerem i przesyła dane o temperaturze powietrza, ciśnieniu atmosferycznym oraz wilgotności powietrza. Odczyt tych parametrów odbywa się w każdej sekundzie, a w związku z tym otrzymujemy dane z częstotliwością 1 Hz. W linii numer 311 odczyt temperatury jest przekazywany do dwóch zmiennych – część całkowita oraz część ułamkowa. Biblioteka dla tego czujnika została odpowiednio zmodyfikowana, aby móc poprawnie wyświetlać temperatury ujemne. W liniach numer 312 – 345 znajduje się kod, którego zadaniem jest wyświetlanie temperatury na wyświetlaczu OLED. Kod ten został przetestowany z różnymi wartościami temperatur. W linii numer 349 wartość ciśnienia atmosferycznego jest przekazywana do dwóch zmiennych – część całkowita oraz część ułamkowa. Część całkowita *hp\_int* jest typu *uint16\_t*, ponieważ wartości ciśnienia atmosferycznego są większe od 255 (maksymalna liczba dla typu *uint8\_t*).

W liniach numer 350 – 362 znajduje się kod, którego zadaniem jest wyświetlanie ciśnienia atmosferycznego na wyświetlaczu OLED. Kod ten został przetestowany z różnymi wartościami omawianego składnika pogody. W linii numer 366 wartość wilgotności powietrza jest przekazywana do dwóch zmiennych – część całkowita oraz część ułamkowa. W liniach numer 367 – 379 został przedstawiony kod, którego zadaniem jest wyświetlanie wilgotności na wyświetlaczu OLED. Kod ten został przetestowany z różnymi wartościami wilgotności powietrza. W przypadku obsługi czujnika BME280, duża część napisanego kodu odpowiada za poprawne wyświetlanie danych na wyświetlaczu OLED – jest to ważne, gdyż interfejs użytkownika powinien być pozbawiony wszystkich błędów związanych między innymi z wyświetlaniem danych.

```

310 //*****DISPLAY TEMPERATURE FROM BME280 SENSOR*****//
311 if( 0 == mkp_read_temp(st_bme280, &t_int, &t_fract) ) // Reading and displaying a temperature in °C
312 { // (-40°C to 85°C) from BME280 sensor on OLED display
313
314     if( t_int == 0 && t_fract < 0 )
315     {
316         t_fract = -t_fract;
317         lcd_gotoxy(0,4);
318         lcd_puts_p(PSTR("Temperature: -"));
319         t_fract_flag = 1;
320     }
321     else if( t_int < 0 && t_fract < 0 )
322     {
323         t_fract = -t_fract;
324         lcd_gotoxy(0,4);
325         lcd_puts_p(PSTR("Temperature: "));
326     }
327     else
328     {
329         lcd_gotoxy(0,4);
330         lcd_puts_p(PSTR("Temperature: "));
331         t_fract_flag = 0;
332     }
333
334     if (t_int <= -10 ) lcd_gotoxy(13,4);
335     else if (t_int > -10 && t_int < 0) lcd_gotoxy(14,4);
336     else if (t_int >= 0 && t_int < 10) lcd_gotoxy(15,4);
337     else if (t_int >= 10 ) lcd_gotoxy(14,4);
338
339     oled_int( t_int );
340     lcd_puts_p(PSTR("."));
341     if( t_fract < 10 ) lcd_puts_p(PSTR("0"));
342     oled_int( t_fract );
343     lcd_puts_p(PSTR("°C"));
344 }
345
346

```

Rys. 4.15. Czujnik środowiskowy BME280 – odczytywanie temperatury powietrza oraz wyświetlanie danych na wyświetlaczu OLED

```

348 //*****DISPLAY PRESSURE FROM BME280 SENSOR*****//
349 if( !mkp_read_pressure( st_bmp280, &hp_int, &hp_fract ) ) // Reading and displaying a pressure in hPa
350 { // (300 hPa to 1100 hPa) from BME280 sensor on OLED display
351     lcd_gotoxy(0,5);
352     lcd_puts_p(PSTR("Pressure: "));
353
354     if(hp_int < 1000) lcd_gotoxy(11,5);
355     else lcd_gotoxy(10,5);
356
357     oled_long_int(hp_int);
358     lcd_puts_p(PSTR("."));
359     if( hp_fract < 10 ) lcd_puts_p(PSTR("0"));
360     oled_int( hp_fract );
361     lcd_puts_p(PSTR(" hPa"));
362 }
363
364
365 //*****DISPLAY HUMIDITY FROM BME280 SENSOR*****//
366 if( !mkp_read_humidity( st_bme280, &hm_int, &hm_fract ) ) // Reading and displaying a humidity in %
367 { // (10% to 100% RH) from BME280 sensor on OLED display
368     lcd_gotoxy(0,6);
369     lcd_puts_p(PSTR("Humidity: "));
370
371     if(hm_int < 10) lcd_gotoxy(15,6);
372     else lcd_gotoxy(14,6);
373
374     oled_int(hm_int);
375     lcd_puts_p(PSTR("."));
376     if( hm_int < 100 && hm_fract < 10) lcd_puts_p(PSTR("0"));
377     oled_int(hm_fract);
378     lcd_puts_p(PSTR(" %"));
379 }
380
381

```

Rys. 4.16. Czujnik środowiskowy BME280 – odczytywanie ciśnienia atmosferycznego, wilgotności powietrza oraz wyświetlanie danych na wyświetlaczu OLED

Fragment kodu zaprezentowany na rysunku 4.17. kończy procesy, które są wykonywane podczas wystąpienia cyklicznego przerwania z układu RTC. W linii numer 383 znajduje się kod, który wyświetla w ostatniej linii wyświetlacza OLED informację, że dane pomiarowe nie są logowane poprzez komunikację UART – ten stan jest domyślny. Gdy opcja logowania danych zostanie włączona, to na wyświetlaczu LED pojawi się poprawny komunikat „DATA send”. W linii numer 385 znajduje się procedura, której zadaniem jest przesłanie danych do sterownika wyświetlacza OLED oraz wyświetlenie na nim danych. W linii numer 390 znajduje się procedura,

której zadaniem jest logowanie danych pomiarowych poprzez komunikację UART. Logowanie to następuje co każdą sekundę – dane są wysyłane, jeśli wartość sekund się zmienia. W związku z tym w linii numer 396 została wprowadzona zmienna pomocnicza `ss_memory`, która przechowuje aktualną wartość sekund pod koniec każdego zdarzenia z układu RTC (linia numer 397). Zmienna znajdująca się w linii numer 393 została wspomniana podczas omawiania układu zegara rzeczywistego i przedstawia ona zerowanie flagi `int0_flag` pod koniec każdego zdarzenia z układu RTC.

```

383         if (uart_logs_data == 0) lcd_gotoxy(0,7), lcd_puts_p(PSTR("DATA don't send")); // information for user about logging data with measurements
384
385         lcd_display(); // display data on OLED display
386
387
388
389         /*****FUNCTIONALITY OF UART DATA LOGGING*****/
390         if (ss_memory != ss && uart_logs_data == 1) uart_data_logging(); // send measurements from sensors via UART every second when logging has been activated
391
392
393         int0_flag = 0; // reset a flag for RTC event
394
395
396         ss_memory = ss; // save actual value of seconds

```

Rys. 4.17. Część procesów, które są wykonywane podczas wystąpienia cyklicznego przerwania z układu RTC

Procedury przerwań oraz definicje zoptymalizowanych funkcji dla tego projektu znajdują się w liniach numer 405 – 901 kodu źródłowego w pliku *main.c*. Procedury przerwań oraz funkcje zostały opisane stosownymi komentarzami tak jak kod główny, aby ich analiza była łatwiejsza.

## 5. OMÓWIENIE ZREALIZOWANYCH BADAŃ ZA POMOCĄ STACJI POMIAROWEJ

W celu sprawdzenia stabilności kodu oraz poprawności pracy urządzenia bez względu na panujące warunki otoczenia, zostały przeprowadzone badania parametrów powietrza w pięciu niezależnych seriach badań. Badania te odbyły się w zróżnicowanych warunkach otoczenia, aby sprawdzić wpływ warunków otoczenia na zaimplementowane czujniki DS18B20 oraz BME280.

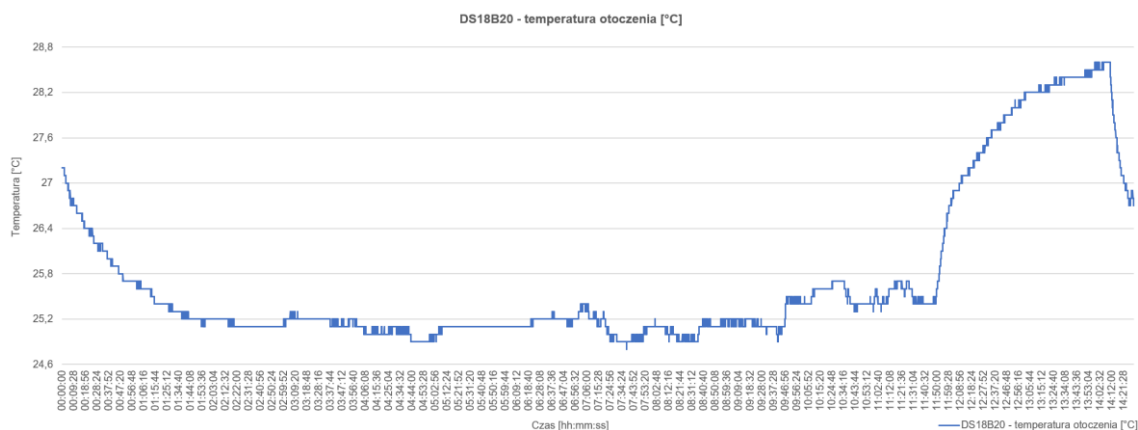
Przetworzone pliki ze stacji pomiarowej zawierają informacje takie jak:

- czas wykonania pomiaru;
- datę wykonania pomiaru;
- temperaturę z czujnika DS18B20 w °C;
- temperaturę z czujnika BME280 w °C;
- $|\Delta T| = |DS18B20 - BME280|$ , czyli moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w °C;
- ciśnienie z czujnika BME280 w hPa;
- wilgotność z czujnika BME280 w %;

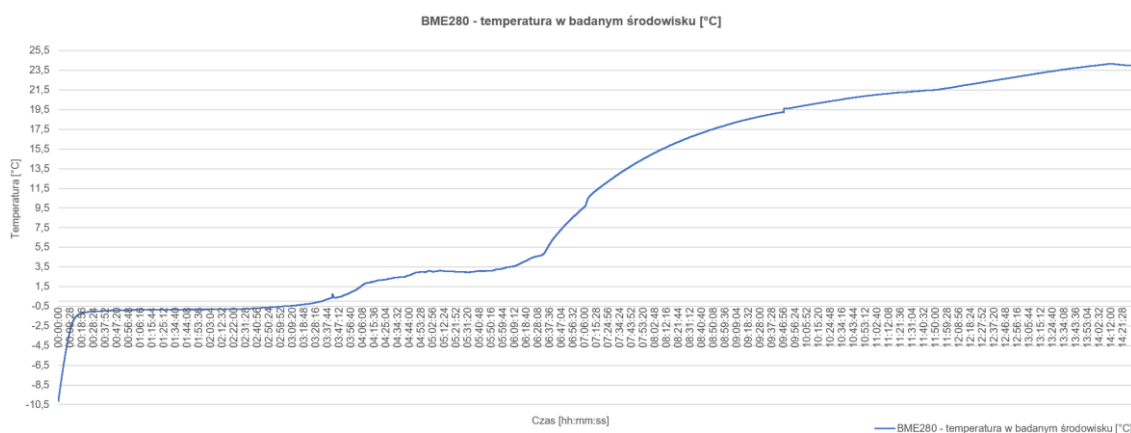
Na podstawie uzyskanych danych, dla każdej serii pomiarowej przedstawionej w kolejnych podrozdziałach, wykonano wykresy przedstawiające wyniki badań oraz tabelę podsumowującą pomiary. Dla każdego badania sformułowano również krótki komentarz podsumowujący.

### 5.1. *Badanie nr 1. Zamarznięta woda w plastikowym kubku o pojemności 250 ml*

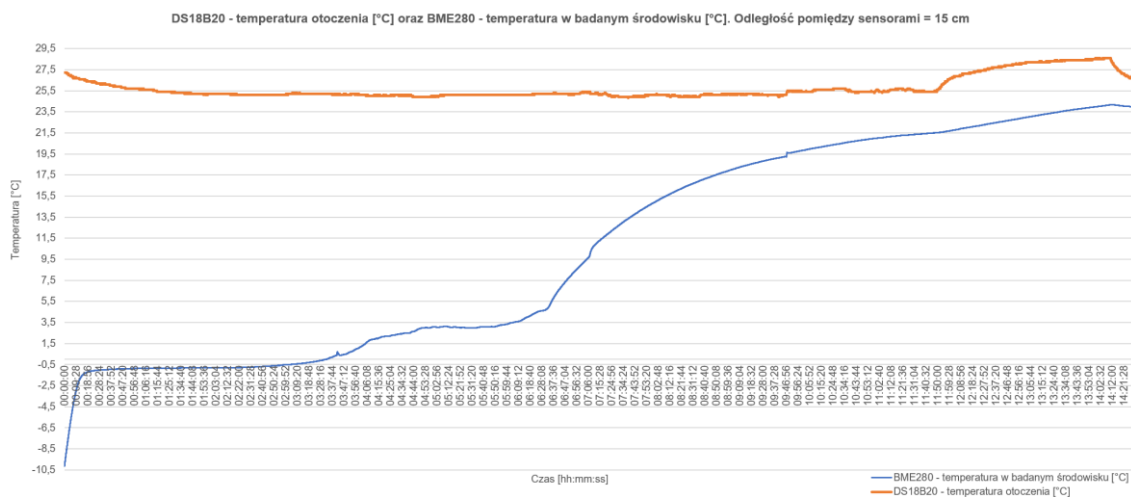
Pierwsza seria pomiarów polegała na zbadaniu zamarzniętej wody w plastikowym kubku o pojemności 250 ml. Czujnik BME280 został umieszczony w kubku w trzech workach strunowych, a następnie został zamrożony w zamrażalniku. Element elektroniczny ze względów bezpieczeństwa nie miał bezpośredniego kontaktu z wodą oraz lodem. Po zakończeniu badań i sprawdzeniu czujnika, był on suchy i nie miał na sobie śladów wilgoci. Cyfrowy czujnik temperatury DS18B20 kontrolował temperaturę panującą w pomieszczeniu, w którym ten pomiar został wykonany. W badaniu tym odległość między czujnikami wynosiła 15 cm. Głównym celem tego badania było sprawdzenie, jak zachowa się czujnik BME280 w ujemnej temperaturze. Temperatura graniczna, czyli -40 °C nie została osiągnięta, natomiast dodatkowo został sprawdzony interfejs komunikacji z użytkownikiem – czyli wyświetlacz OLED oraz komunikacja UART.



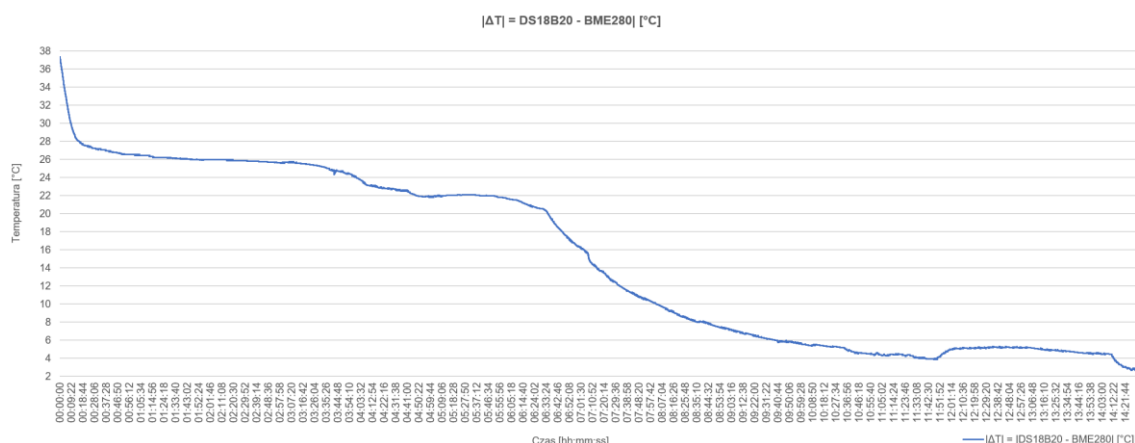
Rys. 5.1. Wykres przedstawiający wartości temperatury otoczenia zarejestrowane za pomocą czujnika DS18B20 w danym przedziale czasowym dla badania nr 1



Rys. 5.2. Wykres przedstawiający wartości temperatury w badanym środowisku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 1



Rys. 5.3. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 1



Rys. 5.4. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 1

Tabela 5.1. Podsumowanie pomiarów dla badania nr 1

Parametr	Wartość minimalna	Wartość średnia	Wartość maksymalna
DS18B20 - temperatura otoczenia [°C]	24,80	25,75	28,60
BME280 - temperatura w badanym środowisku [°C]	-10,15	10,68	24,18
$ \Delta T  =  DS18B20 - BME280  [^{\circ}C]$	2,69	15,06	37,35
	Wartość		
Czas badania [h]	14,5		

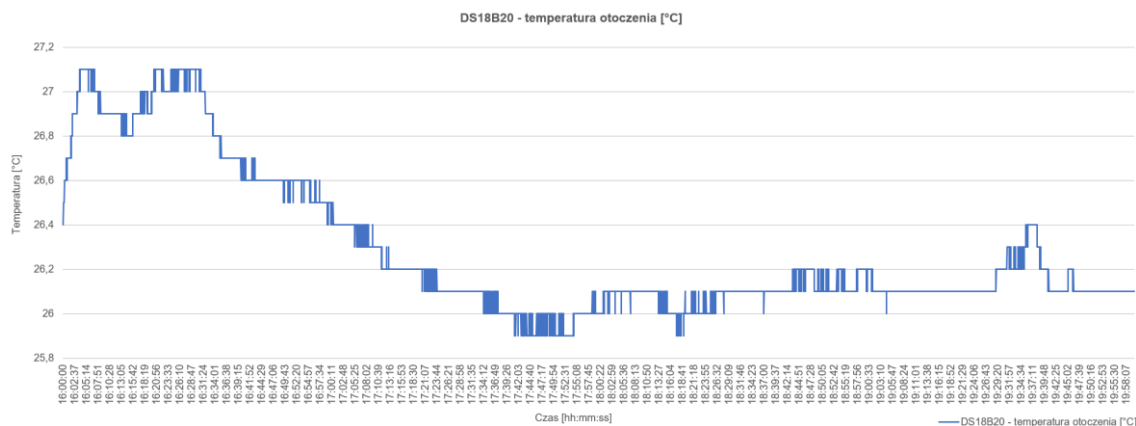
Badanie zamrożonej wody, czyli inaczej mówiąc lodu trwało 14,5 godziny (na podstawie tabeli 5.1.). Na rysunku 5.1. zaobserwowano, że temperatura otoczenia mierzona czujnikiem DS18B20 dynamicznie wzrosła pod koniec badania, co może mieć związek z intensywniejszym korzystaniem z komputera, który znajdował się blisko stacji pomiarowej. Na rysunku 5.2. zauważono, że lód roztopił się po upływie 3,5 godzin, a temperatura wody zaczęła przekraczać 0 °C. Na rysunku 5.3. spostrzeżono, że wraz z upływem czasu temperatura wody się podniosła. Można wywnioskować, że dłuższe badanie doprowadziłoby do wyrównania temperatury wody oraz otoczenia – jest to ukazane na rysunku 5.4. Największy moduł różnicy temperatur między dwoma środowiskami wynosił 37,35 °C.

## 5.2. Badanie nr 2. Gorąca woda w plastikowym kubku o pojemności 250 ml

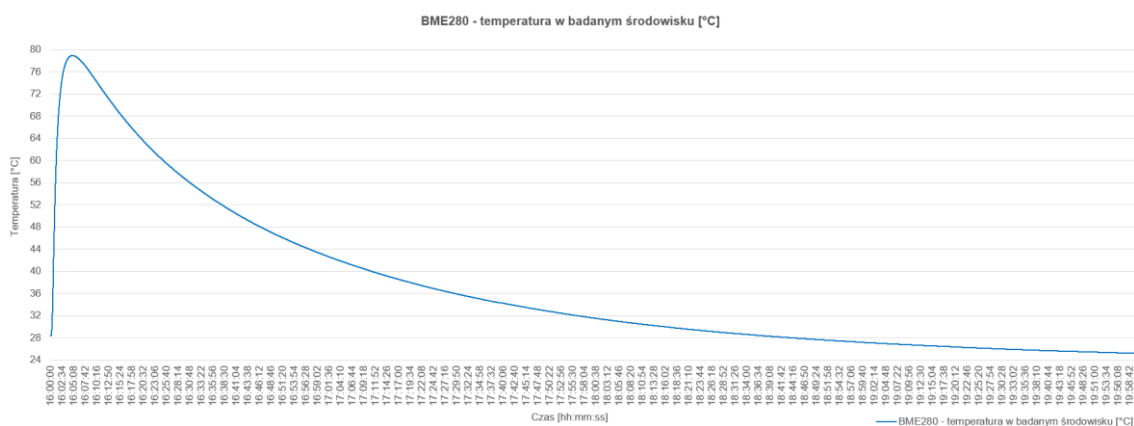
Druga seria pomiarów polegała na zbadaniu gorącej wody w plastikowym kubku o pojemności 250 ml. Czujnik BME280 został umieszczony w kubku w trzech workach strunowych, a następnie został zalany wrzątkiem pochodzącym z czajnika elektrycznego. Za pomocą kamery termowizyjnej INFIRAY T2s+ temperatura wody była na bieżąco monitorowana, a gdy jej wartość była w okolicach 80 °C, pomiary wykonany urządzeniem zostały rozpoczęte. Czujnik BME280 ze względów bezpieczeństwa nie miał bezpośredniego kontaktu z wodą. Po zakończeniu badań i sprawdzeniu czujnika, był on suchy i nie miał na sobie śladów wilgoci. Cyfrowy czujnik temperatury DS18B20 kontrolował temperaturę panującą w pomieszczeniu, w którym ten pomiar został wykonany. W badaniu tym odległość między czujnikami wynosiła 15 cm. Głównym celem tego badania było sprawdzenie, jak zachowa się czujnik BME280 w wysokiej



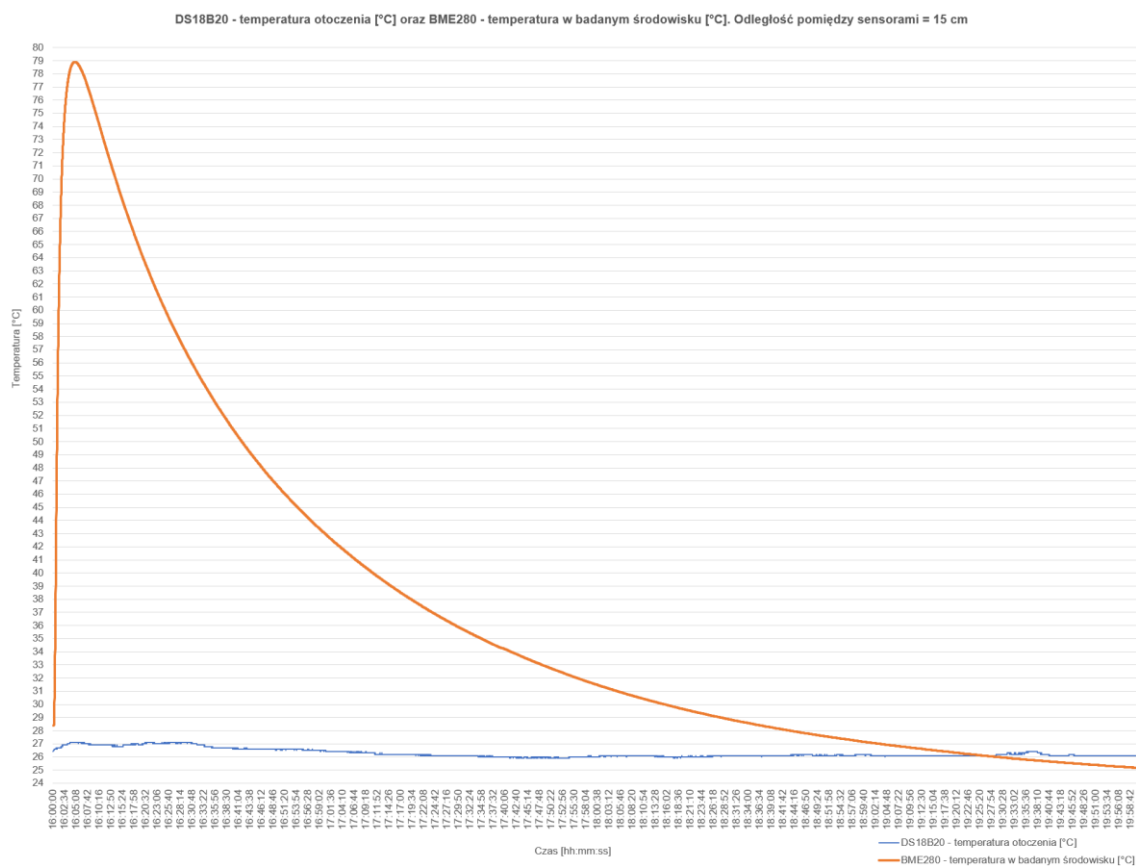
temperaturze. Temperatura graniczna, czyli 85 °C nie została osiągnięta, natomiast dodatkowo został sprawdzony interfejs komunikacji z użytkownikiem – czyli wyświetlacz OLED oraz komunikacja UART.



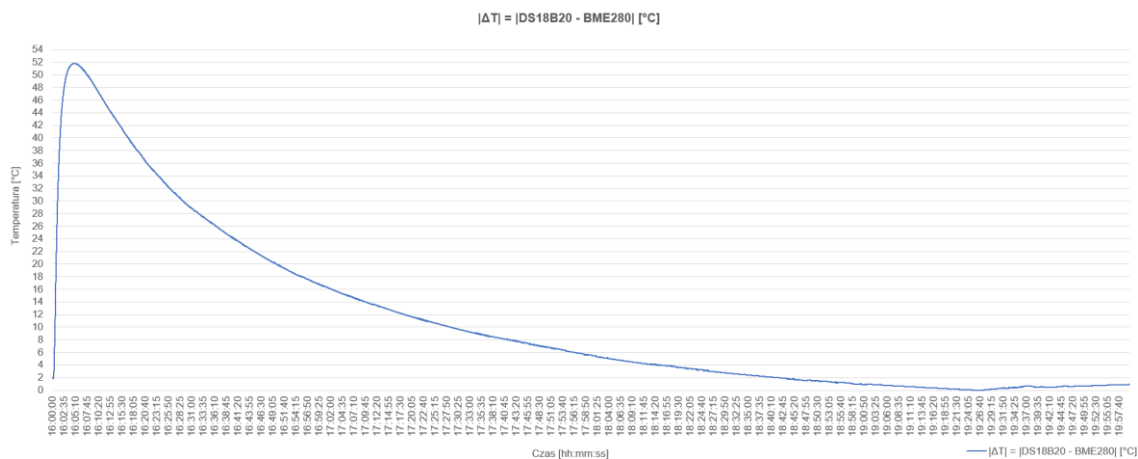
Rys. 5.5. Wykres przedstawiający wartości temperatury otoczenia zarejestrowane za pomocą czujnika DS18B20 w danym przedziale czasowym dla badania nr 2



Rys. 5.6. Wykres przedstawiający wartości temperatury w badanym środowisku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 2



Rys. 5.7. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 2



Rys. 5.8. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 2

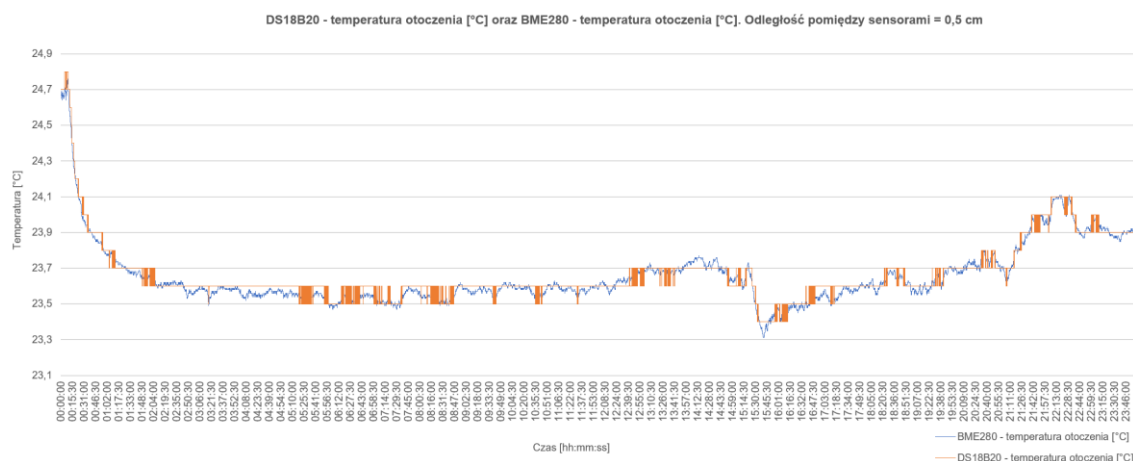
**Tabela 5.2.** Podsumowanie pomiarów dla badania nr 2

Parametr	Wartość minimalna	Wartość średnia	Wartość maksymalna
DS18B20 - temperatura otoczenia [°C]	25,90	26,29	27,10
BME280 - temperatura w badanym środowisku [°C]	25,18	37,33	78,90
$ \Delta T  =  DS18B20 - BME280 $ [°C]	0,00	11,20	51,80
	Wartość		
Czas badania [h]	4		

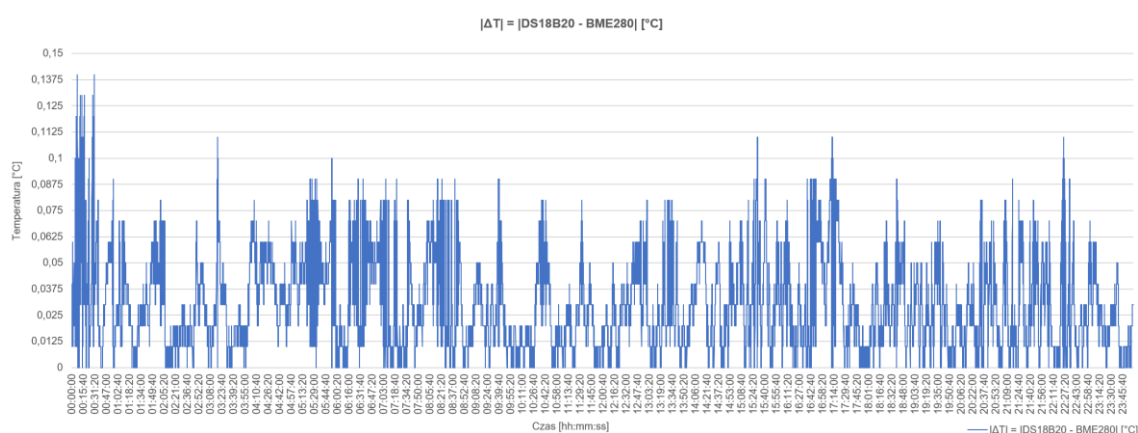
Badanie gorącej wody z czajnika elektrycznego trwało 4 godziny (na podstawie tabeli 5.2.). Na rysunku 5.5. zaobserwowano, że temperatura otoczenia mierzona czujnikiem DS18B20 dynamicznie wzrosła oraz spadła na początku badania, tak jak temperatura mierzona bezpośrednio w plastikowym kubku, co pokazuje rysunek 5.6. Jest to spowodowane ogrzaniem dwóch czujników poprzez gorącą wodę, która znajdowała się w plastikowym kubku. Odległość między dwoma czujnikami nie była odpowiednio duża. Na rysunku 5.7. zauważono, że po 4 godzinach badania temperatura wody nadal wynosiła ponad 25 °C. Na rysunku 5.7. spostrzeżono, że pod koniec badania temperatura otoczenia wzrosła, co może mieć związek z intensywniejszym korzystaniem z komputera, który znajdował się blisko stacji pomiarowej. Na podstawie rysunku 5.8. oraz tabeli numer 2 zauważono, że największy moduł różnicy temperatur między dwoma środowiskami wynosił 51,8 °C.

### **5.3. Badanie nr 3. Porównanie czujników DS18B20 oraz BME280 pod kątem precyzji pomiaru temperatury. Odległość między sensorami $l = 0,5$ cm**

Trzecia seria pomiarów polegała na porównaniu czujników DS18B20 oraz BME280 pod kątem precyzji pomiaru temperatury w pomieszczeniu, gdy odległość między powyższymi sensorami została zminimalizowana. W badaniu tym odległość między czujnikami wynosiła 0,5 cm. Dodatkowo czujnik BME280 monitorował pozostałe parametry w pomieszczeniu, takie jak: ciśnienie atmosferyczne oraz wilgotność powietrza. Głównym celem tego badania było sprawdzenie precyzji pomiaru obu czujników oraz przeanalizowanie, jak zmieniają się warunki panujące w pomieszczeniu w ciągu doby.



Rys. 5.9. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 3



Rys. 5.10. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 3

**Tabela 5.3.** Porównanie pomiarów dla badania nr 3

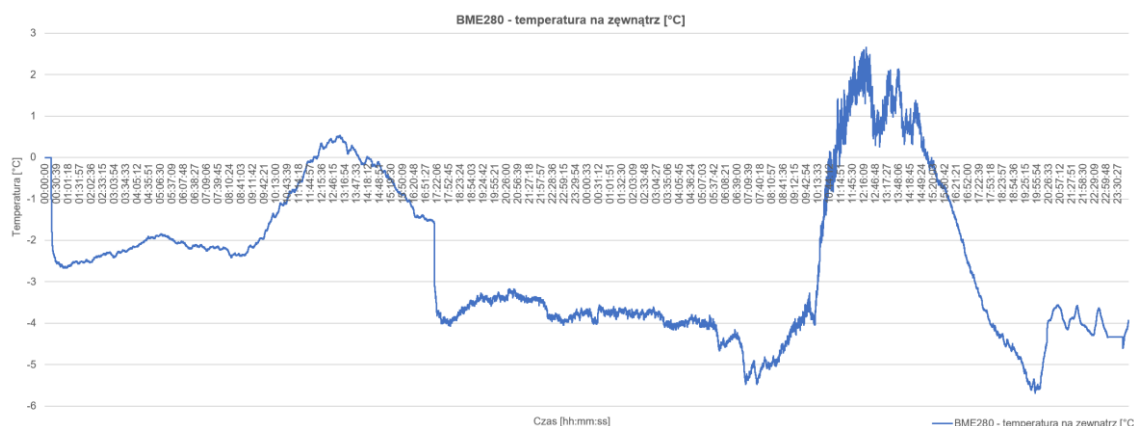
Parametr	Wartość minimalna	Wartość średnia	Wartość maksymalna
DS18B20 - temperatura otoczenia [°C]	23,40	23,67	24,80
BME280 - temperatura otoczenia [°C]	23,31	23,66	24,76
$ \Delta T  =  DS18B20 - BME280  [^{\circ}C]$	0,00	0,03	0,14
	Wartość		
Czas badania [h]	24		

Porównanie czujników DS18B20 oraz BME280 pod kątem precyzji pomiaru temperatury trwało 24 godziny (na podstawie tabeli 5.3.). Na rysunkach 5.9. oraz 5.10. zaobserwowano, że temperatura otoczenia mierzona czujnikiem DS18B20 oraz BME280 rośnie oraz maleje w tym czasie, a dodatkowo wartości te są dobrze odwzorowane. Oznacza to, że oba czujniki dają bardzo zbliżone pomiary oraz urządzenie zostało zweryfikowane pod kątem poprawności wykonywanych pomiarów dla temperatury. Czujniki te znajdowały się w niewielkiej odległości  $l = 0,5$  cm, co ujednoliciło warunki pomiarów dla obu czujników. Podczas tego badania urządzenie pomiarowe znajdowało się w znacznej odległości od komputera, więc wartości na wykresach mogą zostać analizowane pod kątem temperatury otoczenia. Temperatura ta była w większości wykresu

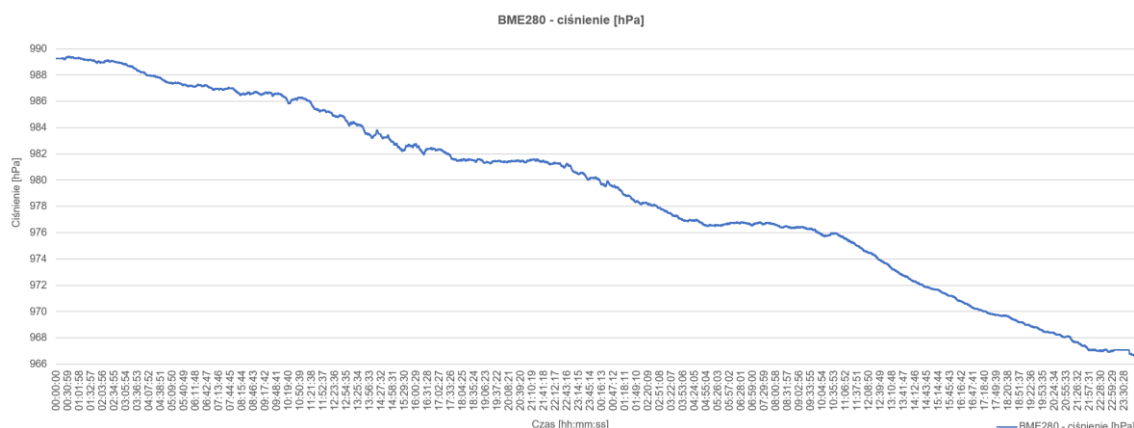
stabilna oraz odpowiednia dla człowieka. Zauważono również, że temperatura w pomieszczeniu została obniżona za pomocą regulatora temperatury tuż przed rozpoczęciem badań. W trakcie cyklu dobowego, w pomieszczeniu nie było dużego ruchu oraz zostało otworzone okno po godzinie 15 – wynika to ze spadku temperatury oraz wilgotności powietrza w mieszkaniu. Na podstawie tabeli 5.3. oraz rysunku 5.10. oszacowano, że średni moduł różnicy temperatur między dwoma czujnikami wynosił 0,03 – wskazuje to na bardzo małe różnice pomiarowe między dwoma czujnikami.

#### 5.4. Badanie nr 4. Badanie warunków atmosferycznych na zewnątrz budynku

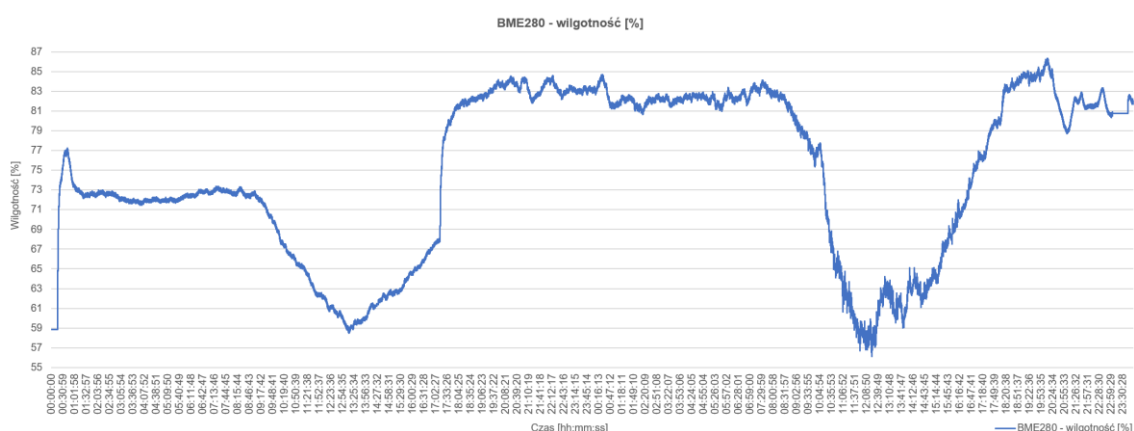
Czwarta seria pomiarów polegała na pomiarze warunków atmosferycznych na zewnątrz budynku za pomocą czujnika BME280. Czujnik BME280 znajdował się na zewnątrz oraz został umieszczony w białej, odbijającej promieniowanie słoneczne obudowie. Był on również chroniony przed wiatrem oraz opadami atmosferycznymi, które mogłyby wpłynąć na wyniki pomiarów. Głównym celem tego badania była analiza warunków atmosferycznych panujących na zewnątrz budynku, a badania były wykonane w miejscowości Garmisch-Partenkirchen. Leży ona w Niemczech w Alpach oraz jest znanym ośrodkiem narciarskim.



Rys. 5.11. Wykres przedstawiający wartości temperatury na zewnątrz budynku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4



Rys. 5.12. Wykres przedstawiający wartości ciśnienia atmosferycznego zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4



Rys. 5.13. Wykres przedstawiający wartości wilgotności powietrza zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4

Tabela 5.4. Podsumowanie pomiarów dla badania nr 4

Parametr	Wartość minimalna	Wartość średnia	Wartość maksymalna
DS18B20 - temperatura otoczenia [°C]	20,60	21,03	22,90
BME280 - temperatura na zewnątrz [°C]	-5,69	-2,50	2,66
$ \Delta T  =  DS18B20 - BME280 $ [°C]	18,44	23,53	26,89
	Wartość		
Czas badania [h]	48		

Pomiary warunków atmosferycznych na zewnątrz budynku były realizowane przez 48 godzin (na podstawie tabeli 5.4.). Na rysunku 5.11. zaobserwowano cykl dobowy, ponieważ wyższe temperatury panują w dzień, natomiast niższe temperatury występują w nocy. Ich niskie wartości wynikają z wykonywania pomiarów w sezonie zimowym. Na rysunku 5.12. zaobserwowano niskie wartości ciśnienia atmosferycznego oraz duży spadek jego wartości o ponad 22 hPa w ciągu 48 godzin. Pomiar ten powinien zostać powtórzony z wykorzystaniem dodatkowego czujnika ciśnienia atmosferycznego. Na podstawie rysunku 5.13. wynioskowano, że wilgotność powietrza poza pomieszczeniem utrzymywała się na bardzo wysokim poziomie przez cały okres badania. Oznacza to że mimo niskich temperatur, w powietrzu znajdowała się duża ilość pary wodnej.

## 6. PODSUMOWANIE

Stacja pomiarowa do analizy parametrów powietrza została zrealizowana zgodnie z założeniami projektowymi. Model stacji pomiarowej jest urządzeniem, które może analizować parametry powietrza w zróżnicowanych środowiskach pomiarowych, a przygotowany kod dla systemów wbudowanych może zostać wykorzystany w wielu aplikacjach. Wykorzystany zestaw uruchomieniowy Atnel ATB 1.05a Andromeda jest dobrą platformą do rozwoju projektów, a dzięki jego zastosowaniu model stacji pomiarowej został wykonany starannie. Niewątpliwą zaletą wykonanego projektu jest jego wielozadaniowość oraz możliwość dalszej rozbudowy o kolejne moduły elektroniczne w zależności od potrzeb użytkownika.

Dane z dwóch czujników DS18B20 oraz BME280 są przetwarzane przez mikrokontroler, a parametry takie jak: temperatura powietrza, ciśnienie atmosferyczne oraz wilgotność powietrza są prezentowane na wyświetlaczu OLED. Dodatkowo stacja pomiarowa wyświetla aktualny czas oraz datę. Podczas dłuższych analiz środowiskowych, użytkownik może skorzystać z zapisywania pomiarów do pliku dzięki zastosowaniu komunikacji UART. Funkcjonalność ta sprawia, że urządzenie staje się w pełni bezobsługowe oraz pozwala na analizę parametrów powietrza w późniejszym czasie. Sterowanie urządzeniem może zostać zrealizowane za pomocą fizycznych przycisków, pilota podczerwieni oraz komunikacji UART. Dzięki temu stacja pomiarowa może zostać umieszczona w trudno dostępnym miejscu oraz nadal możemy mieć pełną kontrolę nad zachodzącymi procesami oraz pomiarami.

Podczas testu urządzenia pod względem niezawodności części sprzętowej oraz programowej, nie zostały znalezione żadne błędy, które mogłyby wpłynąć na komfort użytkownika. Przeprowadzenie badań w różnych środowiskach, potwierdziło, że zespół pomiarowy radzi sobie z trudnymi warunkami otoczenia. Ponadto logowanie danych poprzez komunikację UART działa perfekcyjnie, co informuje o niezawodności kodu. Kod został pozbawiony błędów oraz został w pełni zoptymalizowany dla zastosowanej części sprzętowej. Urządzenie działa bez przerwy w długim okresie i nie napotyka na żadne ukryte problemy, które mogłyby się pojawić w przygotowanym kodzie źródłowym. Otrzymane wyniki za pomocą urządzenia pomiarowego mogą służyć do amatorskich, jak i do profesjonalnych analiz krótko oraz długoterminowych.

Na rynku istnieje wiele rozwiązań, które umożliwiłyby rozwój stworzonego projektu. Stacja pomiarowa mogłaby zostać wyposażona w dodatkowe czujniki, służące do wykrywania stężenia tlenku węgla CO oraz dwutlenku węgla CO<sub>2</sub>. Zastosowanie czujnika do pomiaru współczynnika PM2.5, czyli ilości tzw. pyłu zawieszonego w powietrzu, pozwoliłoby na stworzenie zaawansowanej stacji do pomiaru jakości powietrza. Dodatkowa implementacja modułu GNSS (ang. Global Navigation Satellite System) umożliwiłaby dokonywanie pomiarów z dokładnymi współrzędnymi geograficznymi. Zmodyfikowane urządzenie pomiarowe mogłoby zostać wyposażone w czytnik kart pamięci do zapisu danych oraz w system łączności bezprzewodowej, który byłby w stanie przysyłać dane na większe odległości. Po zastosowaniu takiej ilości modyfikacji, gotowe urządzenie mogłoby stanowić samodzielną jednostkę pomiarową lub pracować jako część bezzałogowego statku powietrznego do analizy jakości powietrza.

## Wykaz literatury

1. Encyklopedia PWN <https://encyklopedia.pwn.pl/haslo/stacja-meteorologiczna;3940081.html>, dostęp na 12.2023.
2. Wikipedia [https://pl.wikipedia.org/wiki/Temperatura\\_powietrza](https://pl.wikipedia.org/wiki/Temperatura_powietrza), dostęp na 12.2023.
3. ZPE <https://zpe.gov.pl/a/cisnienie-atmosferyczne/DCXoPXLfN>, dostęp na 12.2023.
4. Wikipedia [https://pl.wikipedia.org/wiki/Ci%C5%9Bnienie\\_atmosferyczne](https://pl.wikipedia.org/wiki/Ci%C5%9Bnienie_atmosferyczne), dostęp na 12.2023.
5. ZPE <https://zpe.gov.pl/a/przeczytaj/D11iH5XtL>, dostęp na 12.2023.
6. ZPE <https://zpe.gov.pl/a/skad-sie-bierze-wiatr/DzvgBEdHa>, dostęp na 12.2023.
7. ZPE <https://zpe.gov.pl/a/jak-mierzymy-skladniki-pogody/DCwVlvYlg>, dostęp na 12.2023.
8. ZPE <https://zpe.gov.pl/a/opady-i-osady-atmosferyczne/DbVtJY6kg>, dostęp na 12.2023.
9. ZPE <https://zpe.gov.pl/a/jak-mierzymy-skladniki-pogody/DCwVlvYlg>, dostęp na 12.2023.
10. IMGW <https://obserwator.imgw.pl/2022/06/30/modernizacja-stacji-pomiarow-aerologicznych-imgw-pib/>, dostęp na 12.2023.
11. Wikipedia <https://pl.wikipedia.org/wiki/Agrometeorologia>, dostęp na 12.2023.
12. Wikipedia <https://pl.wikipedia.org/wiki/Aktynometria>, dostęp na 12.2023.
13. GoodAIR <https://goodair.pl/blog/temperatura-i-wilgotnosc-w-domu>, dostęp na 12.2023.
14. Obserwator IMGW <https://obserwator.imgw.pl/2022/05/06/o-cisnieniu-atmosferycznym-i-jego-wplywie-na-zdrowie-i-samopoczucie-czlowieka/>, dostęp na 12.2023.
15. Gees <https://gees.com.pl/poradnik/cisnienie-atmosferyczne-a-samopoczucie-jak-pogoda-wplywa-na-nasz-nastroj-n45>, dostęp na 12.2023.
16. Atmel <https://atmel.pl/instrukcja-atb-rev-1-05a.html>, dostęp na 12.2023.
17. Botland <https://botland.com.pl/moduly-avr/8979-atmel-atb-105a-andromeda-zestaw-uruchomieniowy-z-atmega32a-5904422375980.html>, dostęp na 12.2023.
18. Atmel <https://atmel.pl/instrukcja-atb-rev-1-05.html>, dostęp na 12.2023.
19. Microchip [ATmega32A Datasheet \(microchip.com\)](https://www.microchip.com/atmega32a-datasheet), dostęp na 12.2023.
20. Maxim Integrated Products <https://www.analog.com/media/en/technical-documentation/data-sheets/ds1337-ds1337c.pdf>, dostęp na 12.2023.
21. Waveshare [https://botland.com.pl/index.php?controller=attachment&id\\_attachment=1274](https://botland.com.pl/index.php?controller=attachment&id_attachment=1274), dostęp na 12.2023.
22. BOSCH  
<https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf>, dostęp na 12.2023.
23. msalamon <https://sklep.msalamon.pl/produkt/czujnik-cisnienia-temperatury-i-wilgotnosci-bme280-5v/>, dostęp na 12.2023.
24. Maxim Integrated Products <https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf>, dostęp na 12.2023.
25. Botland <https://botland.com.pl/cyfrowe-czujniki-temperatury/165-czujnik-temperatury-ds18b20-cyfrowy-1-wire-tht-5904422366513.html>, dostęp na 12.2023.
26. DatasheetsPDF <https://datasheetspdf.com/pdf-file/1114275/Vishay/TSOP31236/1>, dostęp na 12.2023.
27. Microchip <https://ww1.microchip.com/downloads/en/DeviceDoc/50002787C.pdf>, dostęp na 12.2023.



28. YouTube [https://www.youtube.com/watch?v=xvhGccFKhME&ab\\_channel=tmfmikro](https://www.youtube.com/watch?v=xvhGccFKhME&ab_channel=tmfmikro), dostęp na 12.2023.
29. Microchip [https://ww1.microchip.com/downloads/en/DeviceDoc/ETN36\\_MPLAB%20Snap%20AVR%20Interface%20Modification.pdf](https://ww1.microchip.com/downloads/en/DeviceDoc/ETN36_MPLAB%20Snap%20AVR%20Interface%20Modification.pdf), dostęp na 12.2023.
30. Elektroda <https://www.elektroda.pl/rtvforum/topic3696770.html>, dostęp na 12.2023.
31. Mikrokontrolery AVR Język C Podstawy programowania BLUEBOOK, Mirosław Kardaś
32. Atmel <https://sklep.atmel.pl/pl/p/MK-PRESSURE-HUMIDITY-AVR-Biblioteka-C/282>, dostęp na 12.2023.
33. Atmel [https://sklep.atmel.pl/pl/p/0581\\_0582-MK-I2C-AVR-Biblioteka-C/244](https://sklep.atmel.pl/pl/p/0581_0582-MK-I2C-AVR-Biblioteka-C/244), dostęp na 12.2023.
34. GitHub <https://github.com/Sylaina/oled-display>, dostęp na 12.2023.
35. Microchip <https://www.microchip.com/en-us/tools-resources/develop/microchip-studio>, dostęp na 12.2023.
36. Microchip <https://ww1.microchip.com/downloads/en/DeviceDoc/Microchip-Studio-UserGuide-DS50002718.pdf>, dostęp na 12.2023.
37. YouTube <https://www.youtube.com/playlist?list=PL9B4edd-p2ajbkbFpi47P9PfMtQuV0OwQ>, dostęp na 12.2023.

## **Zawartość załączonego pliku**

W folderze głównym „Praca dyplomowa” znajdują się podfolder oraz plik:

- Praca dyplomowa – folder, który zawiera wykorzystane biblioteki, kod źródłowy oraz projekt, który można uruchomić w Microchip Studio;
- Przedstawienie stacji monitorującej parametry powietrza.mp4 – plik wideo przedstawiający działanie gotowego projektu.

## Wykaz rysunków

Rys. 1.1. Optymalny zakres wilgotności względnej powietrza, a komfort cieplny powietrza [13]	12
Rys. 2.1. Widok ogólny zestawu uruchomieniowego ATB 1.05a Andromeda [17]	15
Rys. 2.2. Opis wyprowadzeń mikrokontrolera ATmega32A z rodziny AVR firmy Atmel w obudowie przewlekanej - DIP40 [19]	21
Rys. 2.3. Ustawienie zworek dla rezonatora kwarcowego w zestawie ATB [18]	22
Rys. 2.4. Ustawienie zworek konfiguracyjnych dla RTC w zestawie ATB [18]	22
Rys. 2.5. Podłączenie zegara RTC w zestawie ATB [18]	22
Rys. 2.6. Podłączenie czujnika temperatury w zestawie ATB [18]	23
Rys. 2.7. Opis pinów programatora MPLAB Snap dla interfejsu ISP/JTAG. Opis pinów mikrokontrolera ATmega32A dla interfejsu JTAG [27]	24
Rys. 3.1. Stacja monitorująca parametry powietrza - opis elementów	25
Rys. 3.2. Stacja monitorująca parametry powietrza - programator/debugger MPLAB Snap	26
Rys. 3.3. Domyślny stan wyświetlacza stacji pomiarowej	27
Rys. 3.4. Stan wyświetlacza stacji pomiarowej, gdy dane są logowane poprzez komunikację UART	28
Rys. 3.5. Stan wyświetlacza stacji pomiarowej, gdy jest ona kontrolowana z wykorzystaniem podczerwieni	28
Rys. 4.1. Dołączenie bibliotek standardowych oraz plików nagłówkowych	34
Rys. 4.2. Fragment kodu źródłowego zawierający definicje preprocesora	34
Rys. 4.3. Definicje typów oraz zmiennych globalnych	35
Rys. 4.4. Definicje funkcji oraz procedur	36
Rys. 4.5. Inicjalizacja kierunku pinów oraz inicjalizacja przerwań dla mikrokontrolera ATmega32A	37
Rys. 4.6. Procedura obsługi przerwania INT0	37
Rys. 4.7. Zerowanie flagi int0_flag pod koniec zdarzenia z układu RTC	37
Rys. 4.8. Procedura obsługi przerwania timera sprzętowego Timer2	38
Rys. 4.9. Inicjalizacja modułów programowych, globalne odblokowanie przerwań, ustawienie czasu oraz daty	38
Rys. 4.10. Funkcja SuperDebounce do obsługi przycisków. Procedura do interakcji z użytkownikiem poprzez komunikację UART oraz procedura odpowiadająca za sterowanie urządzeniem poprzez podczerwień. Kod odpowiadający za miganie diodą kontrolną	40
Rys. 4.11. Odczytywanie danych z pamięci RAM układu RTC oraz obsługa wyświetlacza 7-segmentowego	41
Rys. 4.12. Fragment kodu odpowiedzialny za wyświetlanie czasu oraz daty na wyświetlaczu OLED	42
Rys. 4.13. Fragment kodu odpowiedzialny za obsługę czujnika temperatury DS18B20	43
Rys. 4.14. Procedura, której zadaniem jest wyświetlanie temperatury z czujnika DS18B20 na wyświetlaczu OLED	43
Rys. 4.15. Czujnik środowiskowy BME280 – odczytywanie temperatury powietrza oraz	

wyświetlanie danych na wyświetlaczu OLED .....	44
Rys. 4.16. Czujnik środowiskowy BME280 – odczytywanie ciśnienia atmosferycznego, wilgotności powietrza oraz wyświetlanie danych na wyświetlaczu OLED .....	44
Rys. 4.17. Część procesów, które są wykonywane podczas wystąpienia cyklicznego przerwania z układu RTC .....	45
Rys. 5.1. Wykres przedstawiający wartości temperatury otoczenia zarejestrowane za pomocą czujnika DS18B20 w danym przedziale czasowym dla badania nr 1 .....	47
Rys. 5.2. Wykres przedstawiający wartości temperatury w badanym środowisku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 1.....	47
Rys. 5.3. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 1 .....	47
Rys. 5.4. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 1 .....	48
Rys. 5.5. Wykres przedstawiający wartości temperatury otoczenia zarejestrowane za pomocą czujnika DS18B20 w danym przedziale czasowym dla badania nr 2 .....	49
Rys. 5.6. Wykres przedstawiający wartości temperatury w badanym środowisku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 2.....	49
Rys. 5.7. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 2 .....	50
Rys. 5.8. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 2.....	50
Rys. 5.9. Wykres przedstawiający dwie wartości temperatury zarejestrowane czujnikami DS18B20 oraz BME280 w danym przedziale czasowym dla badania nr 3 .....	52
Rys. 5.10. Wykres przedstawiający moduł różnicy temperatur między czujnikiem DS18B20, a czujnikiem BME280 w danym przedziale czasowym dla badania nr 3.....	52
Rys. 5.11. Wykres przedstawiający wartości temperatury na zewnątrz budynku zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4.....	53
Rys. 5.12. Wykres przedstawiający wartości ciśnienia atmosferycznego zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4 .....	54
Rys. 5.13. Wykres przedstawiający wartości wilgotności powietrza zarejestrowane za pomocą czujnika BME280 w danym przedziale czasowym dla badania nr 4 .....	54
Rys. A.1. Podfolder oraz plik, które znajdują się w folderze głównym „Praca dyplomowa” .....	63
Rys. A.2. Główne okno programu Microchip Studio .....	64
Rys. A.3. Szczegóły projektu w środowisku Microchip .....	65
Rys. A.4. Zmiana częstotliwości taktowania mikrokontrolera w środowisku Microchip .....	65
Rys. A.5. Dodatkowe funkcjonalności w środowisku Microchip, które są niezbędne do pracy z programem .....	65
Rys. A.6. Okno „Solution Explorer”, w którym można znaleźć wszystkie pliki, które są w naszym projekcie .....	66
Rys. B.1. Uruchomienie modułu Data Visualizer z poziomu głównego paska programu .....	67

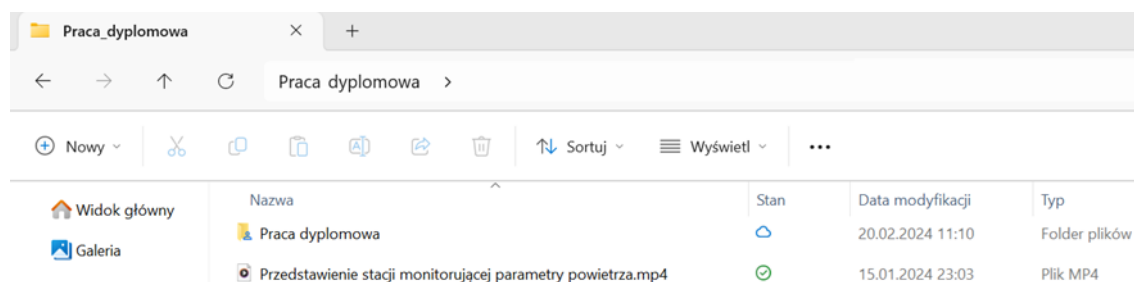
Rys. B.2. Uruchomienie terminala portu szeregowego.....	67
Rys. B.3. Lista dostępnych komend z poziomu terminala portu szeregowego .....	68
Rys. B.4. Zapisywanie w czasie rzeczywistym danych ze stacji pogodowej .....	68

## Wykaz tabel

Tabela 2.1. Lista wykorzystanych komponentów elektronicznych oraz jego opis/specyfikacja techniczna .....	18
Tabela 2.2. Komponenty elektroniczne oraz jego połączenie dla zestawu ATB.....	22
Tabela 3.1. Funkcje realizowane za pomocą fizycznych przycisków.....	29
Tabela 3.2. Funkcje realizowane za pomocą przycisków pilota podświetleni .....	30
Tabela 3.3. Lista dostępnych komend dla komunikacji UART oraz opis realizowanych funkcji..	31
Tabela 5.1. Podsumowanie pomiarów dla badania nr 1 .....	48
Tabela 5.2. Podsumowanie pomiarów dla badania nr 2 .....	51
Tabela 5.3. Porównanie pomiarów dla badania nr 3.....	52
Tabela 5.4. Podsumowanie pomiarów dla badania nr 4 .....	54

## ZAŁĄCZNIK A: Microchip Studio – uruchomienie projektu i omówienie środowiska

Przed uruchomieniem projektu w środowisku Microchip Studio, należy zapoznać się z podfolderem oraz plikiem załączonymi do pracy dyplomowej. Znajdują się one w folderze głównym „Praca dyplomowa”.



Rys. A.1. Podfolder oraz plik, które znajdują się w folderze głównym „Praca dyplomowa”

Na rysunku A.1. przedstawiono podfolder oraz plik, które znajdują się w folderze głównym „Praca dyplomowa”. Można tam znaleźć:

- Praca dyplomowa – folder, który zawiera wykorzystane biblioteki, kod źródłowy oraz projekt, który można uruchomić w Microchip Studio;
- Przedstawienie stacji monitorującej parametry powietrza.mp4 – plik wideo przedstawiający działanie gotowego projektu.

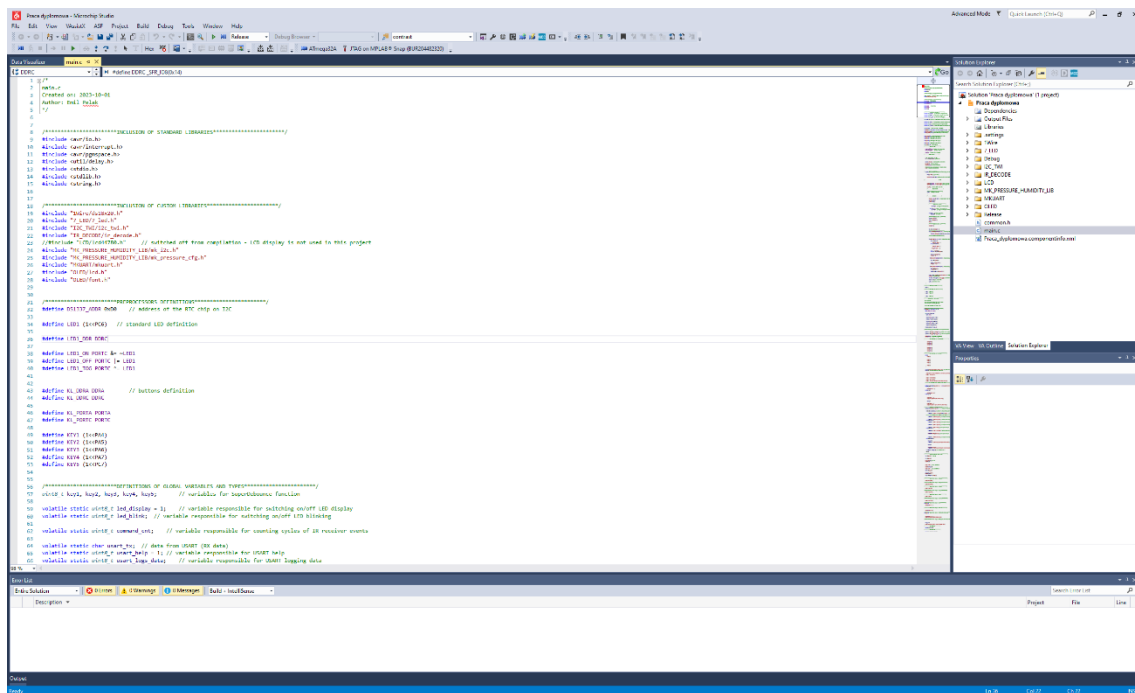
W poprzednich rozdziałach omówiono komponenty elektroniczne, które zostały wykorzystane w moim projekcie. W tym podrozdziale chciałbym omówić stronę programową stacji monitorującej parametry powietrza. Aby uruchomić stworzony przeze mnie projekt, w pierwszej kolejności należy pobrać program *Microchip Studio for AVR and SAM Devices* z oficjalnej strony producenta oprogramowania Microchip [35]. Oprogramowanie można pobrać na dwa sposoby:

- Offline Installer (*pobranie gotowego pliku, który można zainstalować na komputerze bez łącza internetowego*);
- Web Installer (*instalator online, który pobierze wymagane składniki programu – wymagane jest stabilne łącze internetowe*);

Po zainstalowaniu programu i omówieniu folderów oraz plików, można przystąpić do uruchomienia projektu, który znajduje się w dołączonych plikach źródłowych.

Należy otworzyć folder „Praca dyplomowa”, a następnie plik „Praca dyplomowa.atsln”, aby rozpocząć pracę z gotowym projektem.

Po tych czynnościach otworzy się główne okno programu tak jak na rysunku A.2. W głównym oknie programu widoczny jest plik *main.c*, który zawiera główny kod programu napisany w języku C dla systemów wbudowanych oraz moduł „Data Visualizer”.



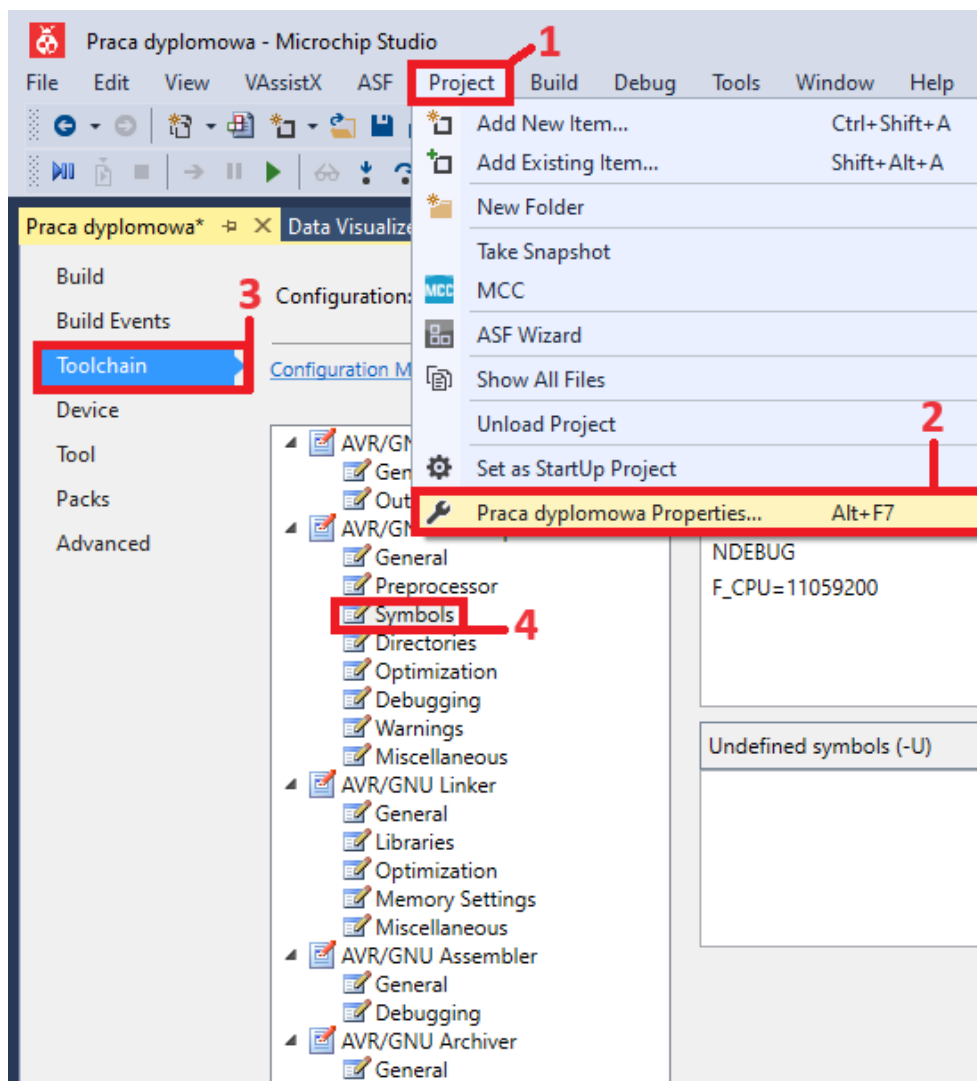
Rys. A.2. Główne okno programu Microchip Studio

Ze względu na rozbudowane funkcje programu Microchip Studio, nie będę omawiać jego wszystkich funkcji, natomiast gotowe poradniki od producenta można znaleźć na oficjalnej stronie Microchip oraz w serwisie YouTube wraz z przedstawieniem wszystkich zagadnień [36, 37].

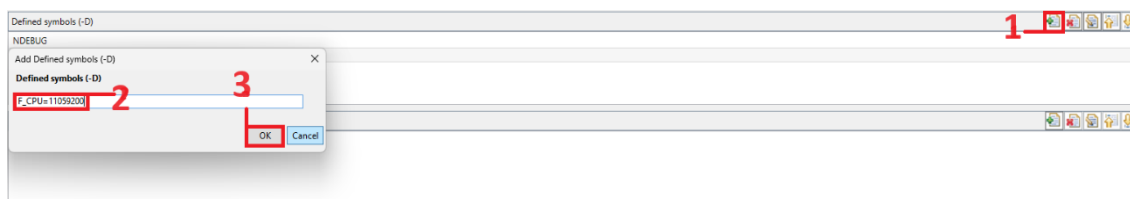
Na rysunku A.3. przedstawiono, jak sprawdzić szczegóły przygotowanego projektu – bardzo ważnym parametrem jest częstotliwość taktowania wykorzystanego mikrokontrolera. W tym celu należy nacisnąć na pasku głównym pozycję „Project”, a następnie „Praca dyplomowa Properties...”. Otworzy się zakładka „Praca dyplomowa” i będziemy mieć kilka pozycji do wyboru. Należy przejść do zakładki „Toolchain”, a następnie należy kliknąć „Symbols”. Opis poszczególnych kroków został również naniesiony na rysunek A.3.

Na rysunku A.4. zostało przedstawione, jak zmienić częstotliwość taktowania mikrokontrolera. Należy kliknąć na przycisk „Add Item”, a następnie wpisujemy w oknie częstotliwość dla użytego rezonatora kwarcowego (w rozważanym przypadku jest to „F\_CPU=11059200”) i wciskamy „OK”. Opis poszczególnych kroków został również naniesiony na rysunek A.4.



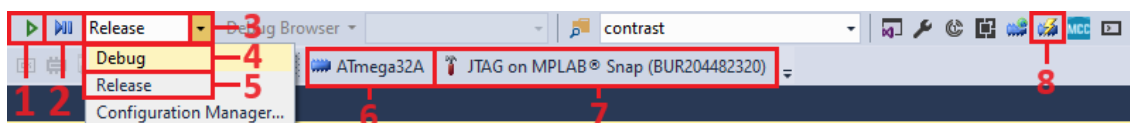


Rys. A.3. Szczegóły projektu w środowisku Microchip



Rys. A.4. Zmiana częstotliwości taktowania mikrokontrolera w środowisku Microchip

Na kolejnym rysunku A.5. przedstawiono ważne funkcjonalności, które są niezbędne do kompilacji/debugowania programu. Pokazane zostało również, jak wgrywać program do układu scalonego oraz jak zmieniać jego parametry pracy.

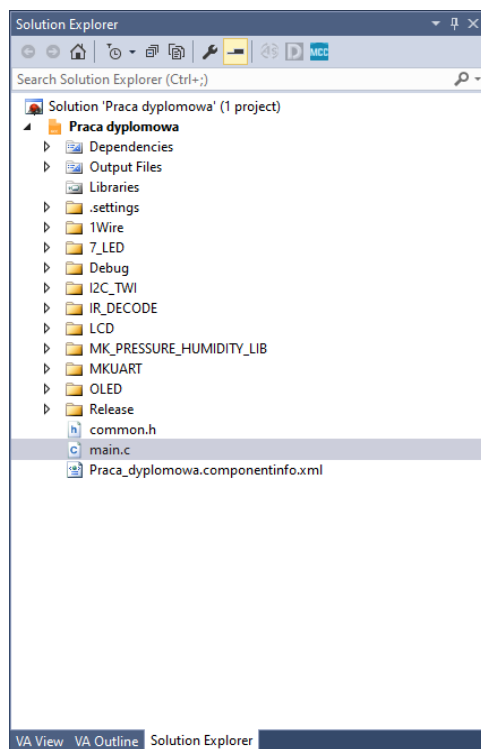


Rys. A.5. Dodatkowe funkcjonalności w środowisku Microchip, które są niezbędne do pracy z programem

Opisy do poszczególnych przycisków (funkcji) oznaczonych na rysunku A.5. znajdują się poniżej:

- „1” – „*Start Without Debugging*” – opcja ta pozwala na wgrywanie programu bezpośrednio do mikrokontrolera;
- „2” – „*Start Debugging and Break*” – funkcjonalność ta umożliwia debugowanie programu;
- „3” – „*Solution Configurations*” – można tutaj wybrać, czy chcemy pracować nad wersją programu do debugowania/gotowej wersji dla mikrokontrolera;
- „4” – „*Debug*” – wybór wersji programu, która jest przeznaczona do debugowania;
- „5” – „*Release*” – wybór wersji programu, która jest znacznie bardziej zoptymalizowana, co skutkuje szybszym i mniejszym kodem. Jest ona trudniejsza do debugowania, ponieważ niektóre instrukcje kodu źródłowego mogą zostać usunięte/przeorganizowane;
- „6” – tą funkcją można sprawdzić specyfikację techniczną użytego mikrokontrolera;
- „7” – omawiana opcja pozwala użytkownikowi na wybór programatora oraz interfejsu komunikacyjnego z mikrokontrolerem;
- „8” – „*Device Programming*” – za pomocą tej opcji można odczytać wszystkie dane z podłączonego mikrokontrolera do programatora. Możemy zaprogramować/odczytać pamięć oraz ustawić/odczytać Fuses i Lock bits.

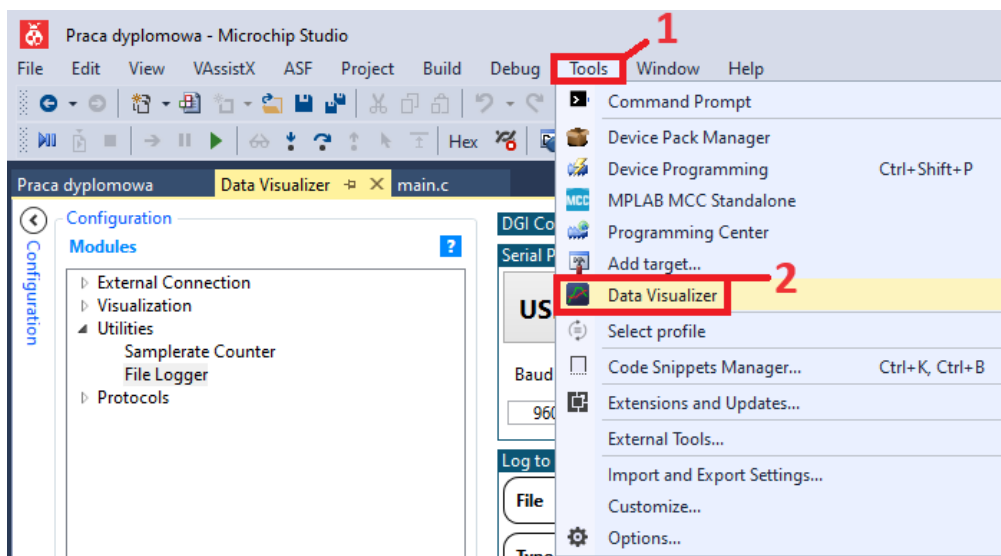
Na poniższym rysunku A.6. znajduje się okno „Solution Explorer”, które znajduje się w prawym górnym rogu programu. Znajdują się tam wszystkie pliki, które są w naszym projekcie. Możemy tam znaleźć między innymi biblioteki dla poszczególnych modułów, kody źródłowe, zmienne oraz funkcje. Takie rozwiązanie jest bardzo praktyczne, ponieważ mamy tam dostęp do wszystkich składników projektu w jednym miejscu oraz możemy poszczególne pliki usuwać, dodawać oraz edytować z poziomu IDE.



Rys. A.6. Okno „Solution Explorer”, w którym można znaleźć wszystkie pliki, które są w naszym projekcie

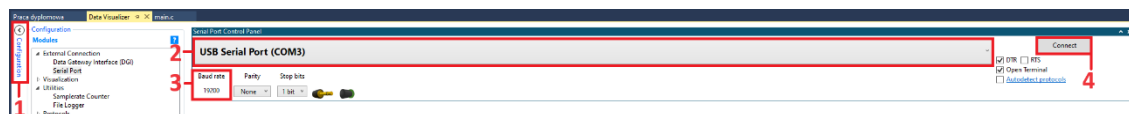
## ZAŁĄCZNIK B: Obsługa stacji pomiarowej oraz logowanie danych pomocą komunikacji UART

Po uruchomieniu projektu widzimy otwarty automatycznie moduł *Data Visualiser*, który umożliwia zbieranie danych ze stacji pomiarowej oraz komunikację z urządzeniem poprzez terminal portu szeregowego. Można go też uruchomić z poziomu głównego paska programu, klikając „Tools”, a następnie „Data Visualizer”. Opis poszczególnych kroków został również naniesiony na rysunek B.1.



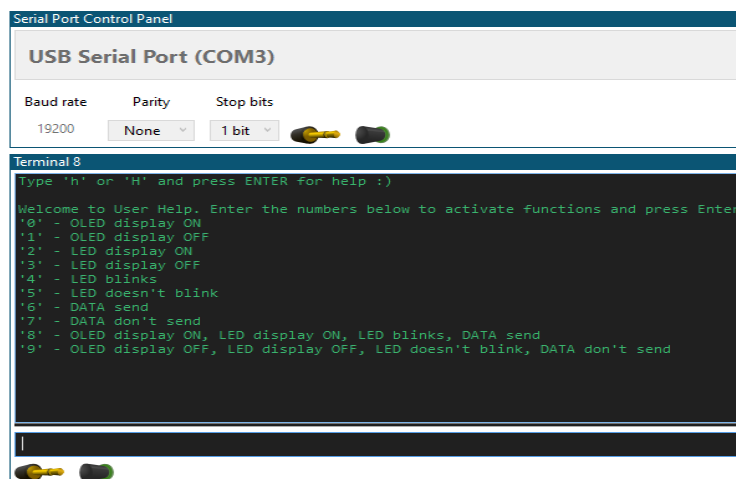
Rys. B.1. Uruchomienie modułu Data Visualizer z poziomu głównego paska programu

Aby korzystać z tego narzędzia, należy kliknąć na zakładkę „Configuration” i wybrać właściwy „Serial Port” dla przejściówki USB/RS232, a następnie wpisać właściwy „Baud Rate” - stacja pogodowa pracuje z prędkością 19200 bitów na sekundę. Następnie należy nacisnąć przycisk „Connect”. Opis poszczególnych kroków został również naniesiony na rysunek B.2.



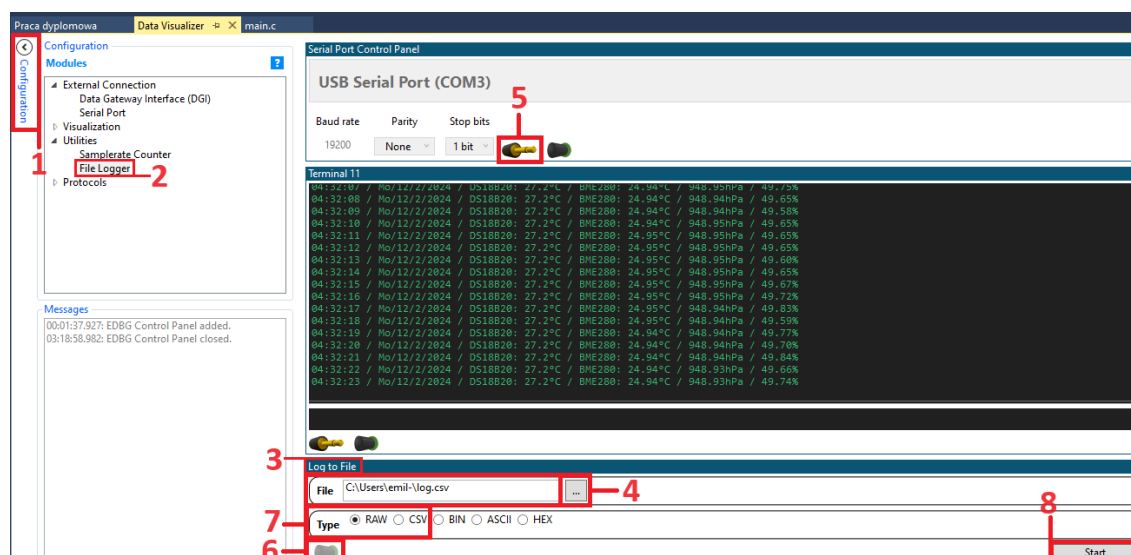
Rys. B.2. Uruchomienie terminala portu szeregowego

Po tych czynnościach, widoczne będzie okno terminala, poprzez które można komunikować się z urządzeniem. Resetując urządzenie, ukaże się informacja „Type 'h' or 'H' and press ENTER for help :)”, czyli „Wpisz „h” lub „H” i naciśnij ENTER, aby uzyskać pomoc”. Po wpisaniu małej lub dużej litery „h”, uzyskamy listę dostępnych komend dla urządzenia. Komendy te można uzyskać z poziomu terminala portu szeregowego w dowolnej chwili, co zostało pokazane na rysunku B.3.



Rys. B.3. Lista dostępnych komend z poziomu terminala portu szeregowego

Aby zapisywać w czasie rzeczywistym dane ze stacji pogodowej do pliku .csv, należy kliknąć na zakładkę „Configuration” i wybrać opcję „File Logger”. Uruchomi się dodatkowy moduł „Log to file”, a w polu „File” można wybrać folder docelowy oraz nazwę pliku. Następnie należy przeciągnąć symbol wtyczki Jack dostępnej przy ustawieniach portu COM do wejścia modułu zbierającego dane. Kolejnym krokiem jest zmiana typu pliku w oknie „Type” z „CSV” na „RAW”. Ostatnim krokiem jest naciśnięcie przycisku „Start”. Aby stacja logowała dane, należy uruchomić logowanie danych za pomocą właściwej komendy. Sterowanie stacją za pomocą przycisków, pilota podczerwieni oraz terminala portu szeregowego również będzie widoczne w logach. Aby mieć pewność, że dane poprawnie się zapisują, należy otworzyć plik docelowy – nie wpłynie to na proces zapisu danych. Opis poszczególnych kroków został również naniesiony na rysunek B.4.



Rys. B.4. Zapisywanie w czasie rzeczywistym danych ze stacji pogodowej