



**AGH**

**AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA  
W KRAKOWIE**

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,  
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

**KATEDRA AUTOMATYKI I INŻYNIERII BIOMEDYCZNEJ**

**Praca inżynierska**

*System pomiaru warunków środowiskowych i  
meteorologicznych*

Autor:

*Michał Mąka*

Kierunek studiów:

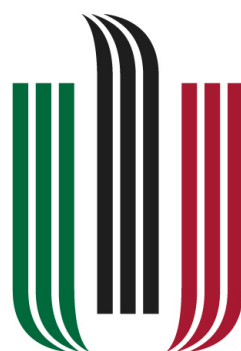
*Automatyka i Robotyka*

Opiekun pracy:

*dr inż. Marek Stencel*

Kraków, 2014

*Oświadczam, świadomy odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonałem osobiście i samodzielnie, i nie korzystałem ze źródeł innych niż wymienione w pracy.*



**AGH**

**AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY  
IN KRAKOW**

**FACULTY OF ELECTRICAL ENGINEERING, AUTOMATICS,  
COMPUTER SCIENCE AND BIOMEDICAL ENGINEERING**

DEPARTMENT OF AUTOMATICS AND BIOENGINEERING

## Engineering Thesis

### *Measurment system of environmental and weather conditions*

Author:	<i>Michał Mąka</i>
Degree programme:	<i>Automatics and Robotics</i>
Supervisor:	<i>Marek Stencel Sc.D</i>

Kraków, 2014



## Spis treści

<b>Wstęp</b> .....	6
<b>1. Mikrokomputer BeagleBone Black</b> .....	9
<b>2. Czujnik ciśnienia atmosferycznego BMP085</b> .....	11
<b>3. Czujnik wilgotności DHT-22</b> .....	15
<b>4. Elektroniczny system pomiarów</b> .....	20
<b>5. Komunikacja w układzie</b> .....	22
5.1. Magistrala szeregową I <sup>2</sup> C.....	22
5.2. Komunikacja z wykorzystaniem GPIO .....	24
<b>6. Programowanie mikrokontrolerów ARM</b> .....	25
6.1. Środowisko programowania .....	25
6.2. Używanie bibliotek Linux'a .....	28
<b>7. Tworzenie aplikacji zbierającej dane pomiarowe</b> .....	29
<b>8. Przechowywanie oraz wyświetlanie wyników</b> .....	33
8.1. Baza danych MySQL.....	33
8.2. Strona internetowa z wynikami pomiarów .....	34
<b>9. Rezultaty pomiarów</b> .....	35
9.1. Warunki zewnętrzne .....	35
9.2. Warunki pokojowe.....	36
<b>10. Podsumowanie</b> .....	38

<b>Bibliografia</b>	<b>39</b>
---------------------	-----------

# Wstęp

## Badanie warunków środowiskowych i meteorologicznych

Zbieranie informacji na temat aktualnych danych pogodowych zawsze było przydatne dla ludzkości, a nawet stało się bardzo ważne w dzisiejszych czasach. Różne przedsiębiorstwa z dziedziny automatyki, budownictwa, przemysłu itp. potrzebują tych danych, aby wiedzieć jak zorganizować swoją pracę oraz przewidzieć plan na najbliższą przyszłość. Posiadając aktualne pomiary pogodowe można zmienić plan pracy, jeżeli warunki atmosferyczne mogą na to nie pozwolić.

Przykładowo przeprowadzenie naprawy sieci elektrycznej przy bardzo dużej wilgotności powietrza jest stosunkowo niebezpieczne, ponieważ mogą występować przepięcia, prowadzące do całkowitego jej uszkodzenia. Badanie temperatury otoczenia jest niezmiennie potrzebne przy instalacjach, które wymagają określonego przedziału ciepła, stosuje się je np. w chłodniach. Dzięki pomiarom temperatury można sterować regulatorem ogrzewania lub chłodzenia w celu uzyskania optymalnych i rekomendowanych warunków pracy urządzeń przemysłowych.

Na podstawie aktualnych danych pogodowych wraz z historią pomiarów, przy zastosowaniu specjalnych algorytmów, można uzyskać z pewnym prawdopodobieństwem prognozę pogody na najbliższą przyszłość. Wyniki przewidywania pogody nigdy nie są idealne, ale pozwalają na przygotowanie się na możliwe warunki atmosferyczne.

Pomiary warunków środowiskowych są nieodłącznym elementem wielu regulatorów automatyki zastosowanych, np. w klimatyzatorach, lodówkach, grzejnikach, nawilżaczach lub osuszaczach powietrza. Na ich podstawie wyliczane jest sterowanie, które następnie jest podawane na element wykonawczy, np. część chłodzącą. Pomiar aktualnej wartości jest nieustannie porównywany z wartością zadaną w cyklu działania urządzenia.

Projekt inteligentnego domu, w którym niemal wszystko jest regulowane automatycznie, wymaga informacji o aktualnych warunkach i na tej podstawie ustawia poziom ogrzewania, nawilżania powietrza itd.

Niniejszy projekt inżynierski ma za zadanie skonstruować w pełni funkcjonalną stację pogodową, która może być wykorzystywana w dowolnych warunkach. Badania meteorologiczne opisane w dalszej części pracy obejmują pomiar temperatury, wilgotności powietrza oraz ciśnienia atmosferycznego.

System pomiarowy ma spełniać następujące wymagania:

- możliwość wykorzystania stacji zarówno wewnątrz jak i na zewnątrz budynku
- konfigurowalność okresu próbkowania
- przechowywanie danych w bazie
- wyświetlanie zmierzonych danych w postaci wykresu oraz aktualnych wartości
- konfigurowalność przedziału wyświetlanych danych

## Plan pracy

Praca składa się z następujących rozdziałów:

Rozdział 1 *Mikrokomputer BeagleBone Black* opisuje główną część układu pomiarowego - mikrokomputer, dzięki któremu możliwe jest uruchomienie całej aplikacji oraz pomiar warunków atmosferycznych. Informuje również o parametrach procesora, dostępnych interfejsach oraz konfiguracji oprogramowania.

Rozdział 2 *Czujnik ciśnienia atmosferycznego BMP085* przedstawia jeden z elementów stacji pogodowej, przy użyciu którego dostarczane są informacje na temat panującego wokół ciśnienia oraz temperatury otoczenia. Dodatkowo opisane zostały cechy charakterystyczne danego urządzenia oraz sposób odczytu pomiarów.

Rozdział 3 *Czujnik wilgotności DHT-22* zajmuje się opisem czujnika wilgotności, opisuje jego dane charakterystyczne, możliwości, sposób komunikacji. Przedstawia również algorytm odczytu danych.

Rozdział 4 *Elektroniczny system pomiarów* jest fragmentem pracy, w którym zaprezentowany jest projekt całego układu pomiarowego, przedstawiony zostaje schemat połączeń pomiędzy mikrokomputerem a czujnikami.

Rozdział 5 *Komunikacja w układzie* przedstawia sposoby komunikacji pomiędzy każdym elementem stacji pogody. Zaprezentowane zostają interfejsy oraz magistrale użyte przy komunikowaniu się czujników z mikrokomputerem.

Rozdział 6 *Programowanie mikrokontrolerów ARM* opisuje w jaki sposób zostaje programowany BeagleBone, zostaje również przedstawione środowisko programowania oraz zalety używania systemu operacyjnego Linux do łatwego wgrywania programu pomiarowego i komunikowania się z czujnikami.

Rozdział 7 *Tworzenie aplikacji zbierającej dane pomiarowe* stanowi o sposobie projektowania programu odczytującego pomiary oraz prezentuje diagram projektowy aplikacji.

Rozdział 8 *Przechowywanie oraz wyświetlanie wyników* przedstawia sposób w jaki dane są gromadzone w bazie danych oraz jak są wyświetlane użytkownikowi chcącemu zobaczyć zmiany parametrów meteorologicznych.

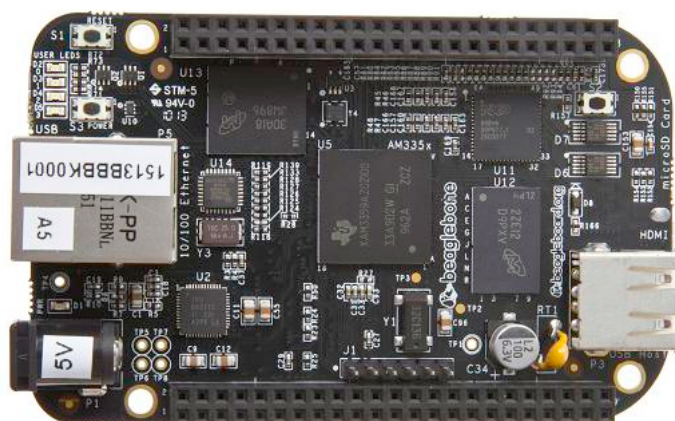
Rozdział 9 *Rezultaty pomiarów* podsumowuje oraz jest odpowiedzialny za testy stworzonego układu pomiarowego wraz z aplikacją. Pokazuje zmierzone dane oraz historię ich zmian w różnych środowiskach.



# 1. Mikrokomputer BeagleBone Black

Projekt inżynierski został zrealizowany, w głównej części, na mikrokomputerze BeagleBone Black. Został on stworzony specjalnie z myślą o programistach OpenSource oraz tych, dla których liczy się niskie zużycie energii. Jest to oparta na procesorze AM335x ARM Cortex-A8, taktowanego częstotliwością 1 GHz, płytki developerska, która została wyposażona w 512 MB pamięci RAM, 2 GB pamięci FLASH, akcelerator grafiki 3D. Posiada szereg różnych interfejsów, takich jak: HDMI, USB, Ethernet, czytnik kart microSD. BeagleBone'a można zasilać na dwa sposoby, pierwszy - poprzez kabel USB podłączony do USB (5V) albo przy użyciu zewnętrznego zasilacza, również 5V. Dla użytkownika zostało również wyprowadzone 96 pinów typu wejście/wyjście.

Na mikrokomputerze można zainstalować i ze swobodą korzystać z najpopularniejszych dystrybucji Linuxa, np. Ubuntu, Debian, Fedora, Arch. Istnieje również możliwość uruchomienia na nim systemu Android.

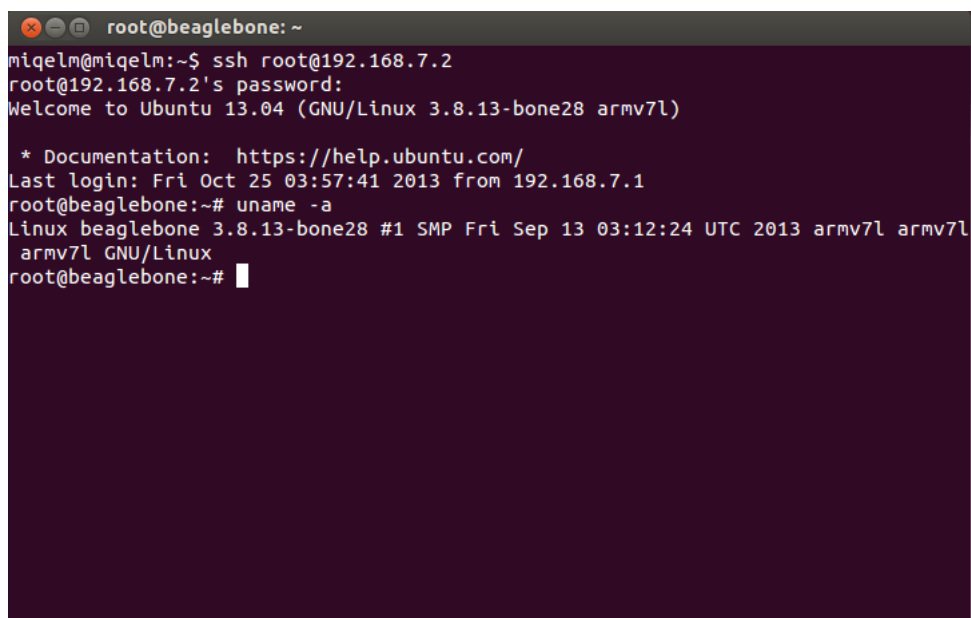


Rysunek 1.1. BeagleBone Black

Po zakupie, domyślnie BeagleBone posiada zainstalowaną dystrybucję Linuxa - Ångström. Z uwagi na większą znajomość innego systemu operacyjnego - Ubuntu, na mikrokomputerze

została zainstalowana właśnie ta dystrybucja w wersji 13.04, dedykowana na platformę ARM Hard Float. Platforma ta posiada zaimplementowaną sprzętową obsługę liczb zmiennoprzecinkowych. Obraz systemu operacyjnego oraz instrukcję jego zainstalowania za pomocą karty pamięci microSD można odnaleźć na głównej stronie projektu ARM HF: [www.armhf.com](http://www.armhf.com).

BeagleBone został podłączony do komputera z zainstalowanym środowiskiem programistycznym przy użyciu portu USB. Po zainstalowaniu odpowiednich sterowników, które znajdują się na oficjalnej stronie producenta tej płytki oraz pamięci FLASH, port ten jest wykrywany jako interfejs sieciowy i tworzona jest sieć łącząca komputer z urządzeniem. Domyślne ustawienia sprawiają, że do BeagleBone'a można podłączyć się przy użyciu protokołu SSH, łącząc się z adresem: 192.168.7.2. Po poprawnym zalogowaniu się do płytki poprzez program *ssh*, dostępny na Ubuntu, otrzymamy ekran podobny do tego poniżej:



```
root@beaglebone: ~
miqelm@miqelm:~$ ssh root@192.168.7.2
root@192.168.7.2's password:
Welcome to Ubuntu 13.04 (GNU/Linux 3.8.13-bone28 armv7l)

 * Documentation:  https://help.ubuntu.com/
Last login: Fri Oct 25 03:57:41 2013 from 192.168.7.1
root@beaglebone:~# uname -a
Linux beaglebone 3.8.13-bone28 #1 SMP Fri Sep 13 03:12:24 UTC 2013 armv7l armv7l
armv7l GNU/Linux
root@beaglebone:~#
```

Rysunek 1.2. Zrzut ekranu z konsoli

## 2. Czujnik ciśnienia atmosferycznego BMP085

### Opis

Czujnik ten jest wysoce precyzyjnym, cyfrowym czujnikiem ciśnienia oraz temperatury powietrza. Jego energooszczędność oraz niskie zapotrzebowanie na zasilanie napięciem (do 3,6 V) sprawia, że jest idealnym rozwiązaniem do zastosowań w urządzeniach mobilnych codziennego użytku, tak jak telefony komórkowe, nawigacje GPS itd. Dzięki bardzo szybkiemu czasowi przetwarzania danych (do 7,5 ms) oraz niskiemu wpływowi zakłóceń daje rezultaty, niewiele różniące się od warunków mierzonych.

BMP085 jest często używany w prognozowaniu pogody, ze względu na obecność dwóch czujników (ciśnienia i temperatury). Można go również wykorzystać do wyznaczenia wysokości względnej oraz bezwzględnej od poziomu morza. W tym celu używane jest prawo zależności wysokości od ciśnienia, tzw. wzór barometryczny. Po uproszczeniu wygląda on następująco:

$$wysokosc = 44330 * (1 - (\frac{p}{p_0})^{\frac{1}{5,255}})$$

$p$  - ciśnienie na badanej wysokości

$p_0$  - ciśnienie na poziomie odniesienia

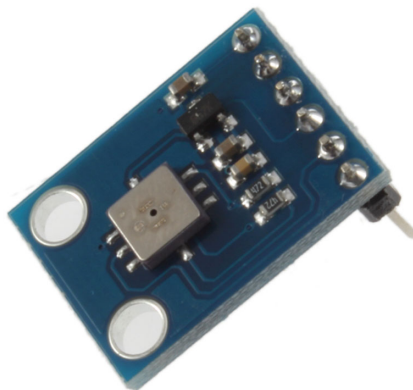
Znając wartość ciśnienia na poziomie odniesienia, istnieje możliwość wyznaczenia względnej wysokości ciała nad znaną wysokością bazową. Taka własność ciśnienia jest używana na lotniskach, samolot w momencie lądowania jest informowany o ciśnieniu panującym na lotnisku, piloci znają również ciśnienia panujące wokół samolotu. Na tej podstawie wyznaczana jest względna wysokość maszyny od poziomu pasa startowego.

### Dane charakterystyczne

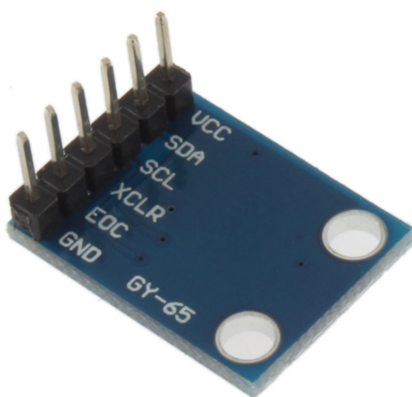
Czujnik BMP085, zgodnie ze specyfikacją, może pracować bezpiecznie w przedziale temperatur od -40 do +85 °C. Rozdzielczość pomiarów dla ciśnienia atmosferycznego wynosi 0.01

hPa, natomiast dla temperatury jest to 0.1 °C. Dokładność badania ciśnienia to maksymalnie +- 4 hPa, dla temperatury to +- 2°C.

Zdjęcia 2.1 oraz 2.2 prezentują wygląd czujnika wbudowanego w gotowy układ, z wyprowadzonymi złączami do komunikacji z mikrokontrolerami. Taki moduł został wykorzystany w projekcie inżynierskim.



Rysunek 2.1. Czujnik ciśnienia BMP085 - widok z dołu



Rysunek 2.2. Czujnik ciśnienia BMP085 - widok z góry

Pomiar temperatury oraz ciśnienia jest kompensowany przy użyciu danych kalibracyjnych, które przechowywane są w pamięci EEPROM czujnika.

## Pobieranie danych z czujnika

BMP085 jest przystosowany do łączności za pomocą magistrali  $I^2C$  (magistrala ta zostanie szerzej omówiona w późniejszych rozdziałach). Potrzebuje on do poprawnego działania czterech przewodów: zasilania, uziemienia, linii danych SDA oraz linii zegara SCL. Linie SCL i SDA wymagają podpięcia rezystora podciągającego do zasilania o oporze  $4,7\text{ k}\Omega$ . Zastosowany w pracy czujnik jest zbudowany na gotowej płycie drukowanej z podpiętymi już rezystorami podciągającymi, dzięki temu jest on od razu przygotowany do podpięcia kablami, bez dodatkowego montowania układu.

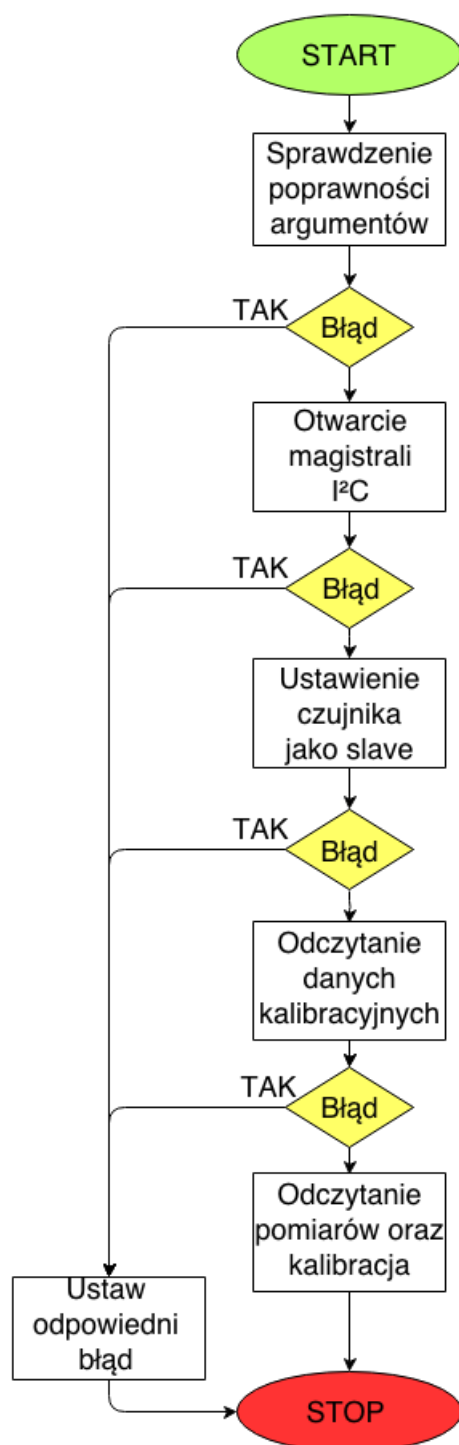
Czujnik podpięty do magistrali  $I^2C$  otrzymuje adres, pod którym on się znajduje w strukturze. BMP085 domyślnie posiada adres 0x68 i przy użyciu tego adresu można odczytywać oraz zapisywać do niego dane.

## Algorytm odczytu pomiarów

Cały algorytm służący do odczytania aktualnych wartości warunków meteorologicznych wymaga odczytania rejestrów czujnika BMP085 za pośrednictwem magistrali  $I^2C$ . Z poziomu języka C oraz systemu Linux najpierw należy otworzyć plik, który jest reprezentacją magistrali i na nim wykonywać operacje odczytu oraz zapisu. Kolejną czynnością jaką należy wykonać jest oznajmienie systemowi, że pod konkretnym adresem znajduje się czujnik i jest on uważany za urządzenie typu *slave*, czyli typu podrzędnego - słucha komend mikrokomputera. Następnie należy odczytać z pamięci EEPROM czujnika dane kalibracyjne, które zostają później wykorzystane do przeliczenia poprawnych wartości pomiarów.

Specyfikacja czujnika jasno określa kolejne kroki, jakie należy wykonać, aby przejść od odczytania danych kalibracyjnych do wyznaczenia pomiarów. Są to specjalnie stworzone wzory, które z wartości bajtów otrzymanych z czujnika zwracają potrzebne dane.

Diagram przedstawiony na rysunku 2.3 pokazuje algorytm odczytu pomiarów czujnika BMP085.



Rysunek 2.3. Diagram odczytu danych z czujnika BMP085

### 3. Czujnik wilgotności DHT-22

#### Opis

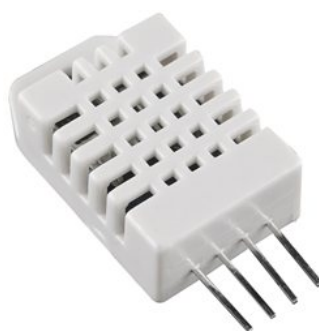
DHT-22 jest cyfrowym oraz fabrycznie skalibrowanym czujnikiem względnej wilgotności i temperatury. Został stworzony w oparciu o 8 bitowy mikrokontroler, do którego podłączone są sensory. Małe rozmiary, stabilność pomiarowa, możliwość zainstalowania czujnika na duże odległości oraz niewielki pobór energii czyni go bardzo dobrym rozwiązaniem, nawet w ciężkim warunkach atmosferycznych.

Czujnik posiada swój własny sposób uzyskiwania pomiarów. Jest on podobny do magistrali szeregowej o nazwie 1-Wire. Do prawidłowego działania urządzenia potrzebne jest zasilanie, linia danych, poprzez którą następuje komunikacja oraz uziemienie. Należy również podłączyć do linii danych oraz zasilania rezystor podciągający o oporze około 3,3 k $\Omega$ .

#### Dane charakterystyczne

Jak podaje specyfikacja czujnika, może być on zasilany napięciem od 3,3 do 6 V, posiada następujące zdolności pomiarowe: wilgotność w zakresie 0 do 100 % (z dokładnością  $\pm 2\%$ ) oraz temperatury od -40 do 80 stopni Celsjusza ( $\pm 0,5$  stopnia).

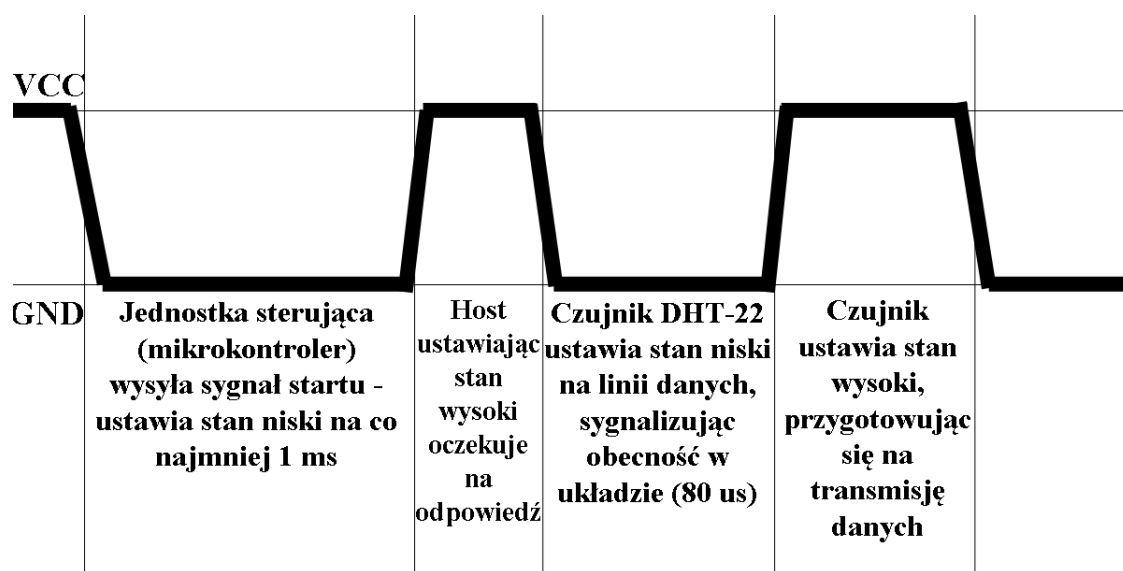
Wygląd czujnika został przedstawiony na poniższym zdjęciu:



Rysunek 3.1. Czujnik wilgotności DHT-22

## Pobieranie danych z czujnika

Zgodnie ze specyfikacją, załączoną do DHT-22, linia danych jest wolna, jeżeli jest na niej ustawiony stan wysoki. Czujnik, gdy nie jest używany, jest ustawiony w tryb spoczynku i niskiego poboru prądu. Poprzez wysłanie sygnału startu jest on wybudzany do trybu pracy i wysyła dane. Aby rozpocząć pomiar trzeba wysłać sygnał startu, należy to uczynić poprzez zwarcie do masy linii danych na co najmniej 1 ms. Po tym czasie czujnik zgłasza swoją obecność w układzie poprzez wystawienie logicznej jedynki na około 20-40  $\mu\text{s}$ , następnie linia danych jest zwalniana oraz ponownie podciągana do zasilania, każda z tych czynności trwa 80  $\mu\text{s}$ . Rysunek 3.2 przedstawia inicjalizację czujnika DHT-22. Po zgłoszeniu swojej obecności w układzie,



Rysunek 3.2. Inicjalizacja czujnika

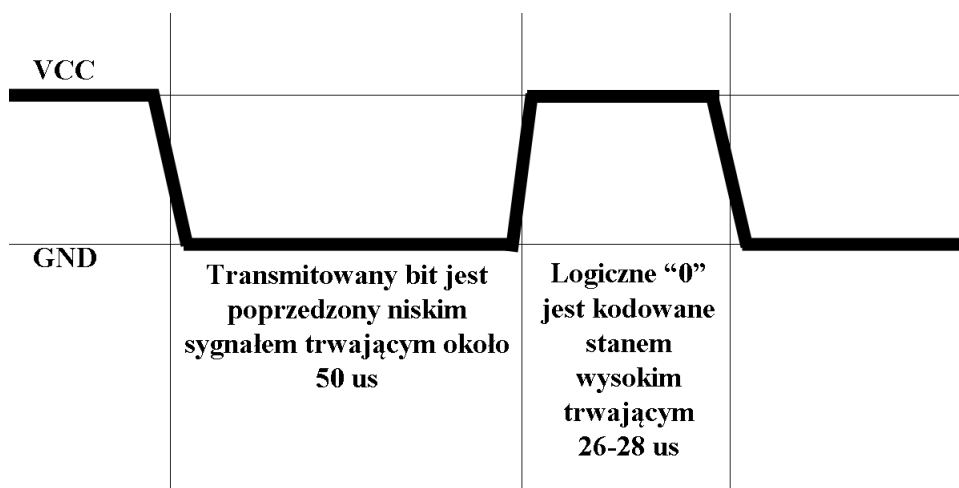
czujnik zaczyna wysyłać 40 bitów danych, pierwsze 16 bitów odpowiada za ciśnienie, następnie 16 to temperatura oraz końcowe 8 bitów służy jako suma kontrolna.

Aby przekonwertować odebrane dane na wskazania czujnika, należy obie liczby 16-bitowe, odpowiadające wilgotności oraz temperaturze, podzielić przez 10. W ten sposób otrzymujemy pomiary z dokładnością do jednego miejsca po przecinku. Po odebraniu danych należy przede wszystkim sprawdzić, czy to co zostało otrzymane jest prawidłowe. Należy pierwsze 4 bajty dodać do siebie oraz sprawdzić, czy 8 młodszych bitów tej wartości są równe ostatniemu odebranemu bajtowi. Jeżeli tak, procedura odebrania została zakończona sukcesem. W przeciwnym wypadku należy spróbować ponownie pobrać dane z czujnika. Próby odczytu należy wykonywać do momentu przekroczenia maksymalnej liczby prób, przez co można uniknąć nieskończonym próbom odczytu.

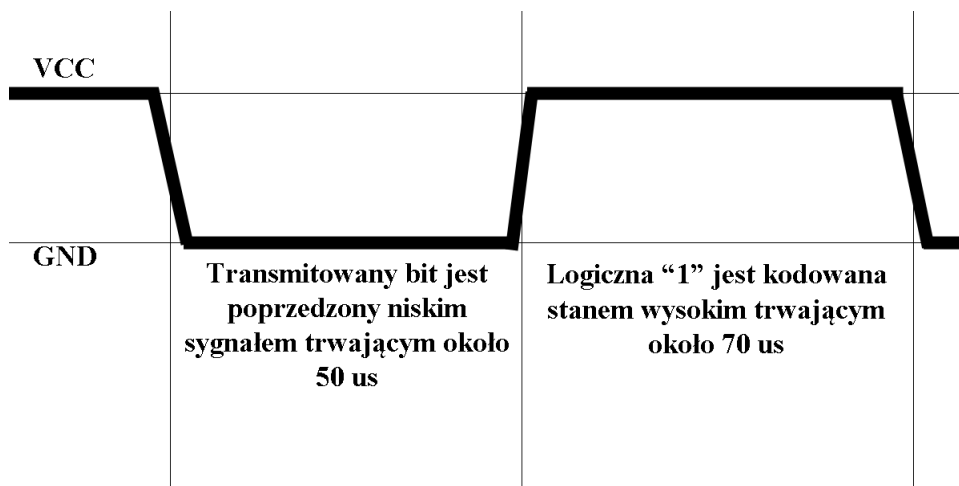


Sposób kodowania informacji na linii danych przez DHT-22 jest następujący: każda wartość logiczna (0 oraz 1) jest reprezentowana przez podpięcie linii do zasilania na ustaloną chwilę czasową. Bit o wartości 0 jest wysyłany jako wysoki poziom przez okres 26-28  $\mu\text{s}$ , natomiast czas trwania sygnału wysokiego dla bitu 1 wynosi 70  $\mu\text{s}$ . Pomiędzy nadaniem odpowiedniego bitu, występuje przerwa - podłączenie linii danych do uziemienia na około 50  $\mu\text{s}$ .

Przebiegi poniżej pokazują wysyłanie danych:



Rysunek 3.3. Wysyłanie logicznego "0"



Rysunek 3.4. Wysyłanie logicznej "1"

Wysyłanie danych jest zakończone, jeżeli linia danych jest podłączona do zasilania oraz nie zmienia się jej stan. Czujnik wtedy przechodzi w stan uśpienia i będzie w nim przebywał aż do pojawienia się następnego sygnału startu. Jeżeli sygnał na linii danych jest zawsze w stanie wysokim, oznacza to niepoprawne działanie czujnika, może to być spowodowane wadliwym podłączeniem.

## Algorytm odczytu pomiarów

Podczas odczytywania danych bardzo ważne są czasy występowania odpowiednich wartości logicznych. Błędna analiza czasu skutkuje złym zinterpretowaniem danych, co prowadzi do sfalszowania wyników.

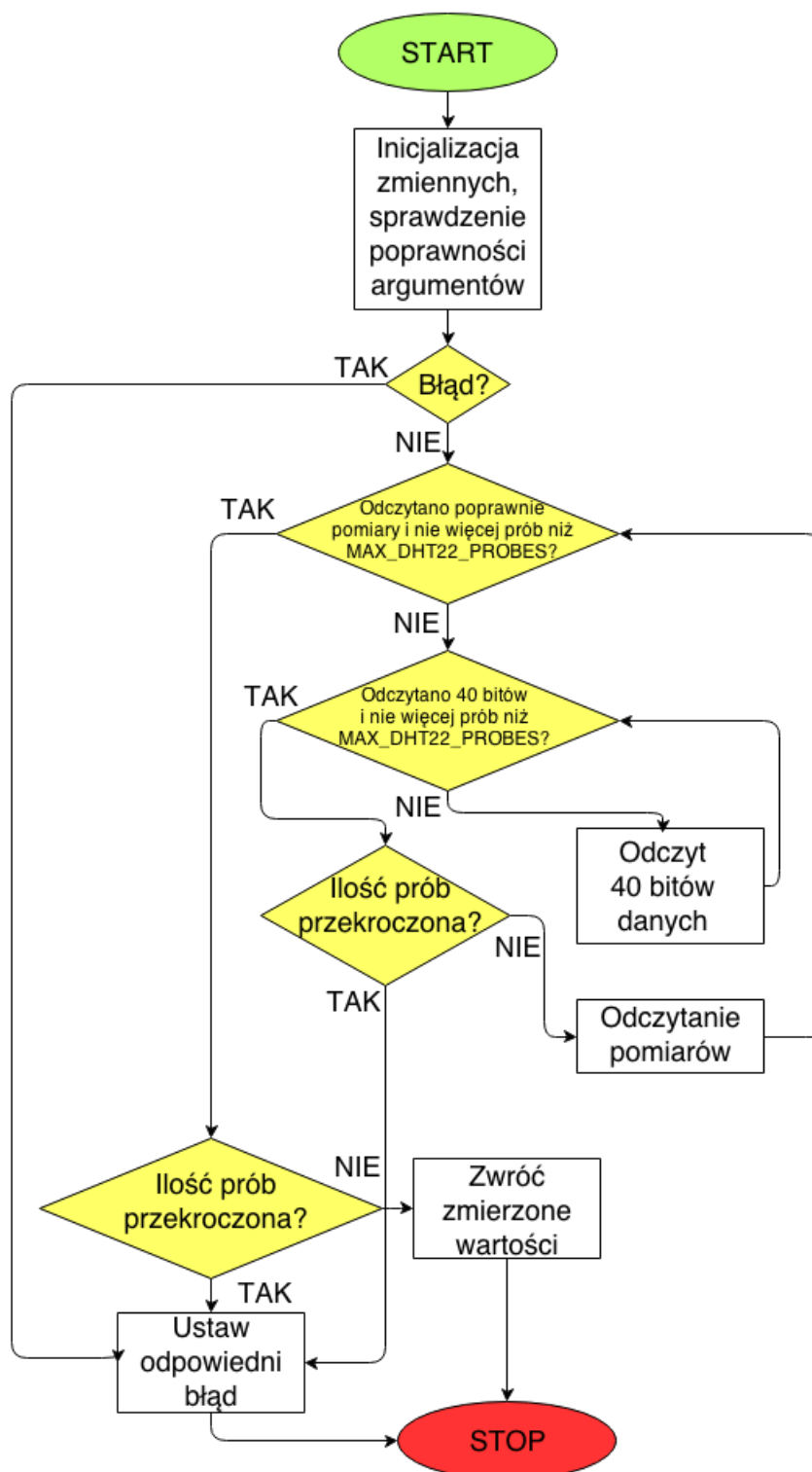
Algorytm zastosowany przy tworzeniu projektu inżynierskiego jest bardzo prosty. Polega on na odczytywaniu przy każdej możliwości stanu linii danych oraz zliczaniu ilości wystąpień stanu wysokiego, gdyż tylko ten koduje wysyłane wartości z czujnika.

Linia danych jest próbkowana do momentu odczytania 40 bitów. Jest ona ustawiana wówczas w stan wysoki, czujnik przechodzi w stan spoczynku. Po zakończeniu odczytu następuje analiza przetworzonych danych.

Ilości wystąpień każdego stanu wysokiego podlegają binaryzacji, czyli procesowi podziału na dwa zbiory. Zadaniem tej funkcjonalności jest znalezienie odpowiedniej wartości granicznej, takiej która jednoznacznie wyznacza ile razy musiał być spróbkowany sygnał na linii danych, aby został on zinterpretowany jako bit "1" lub "0". Zostało to zaimplementowane poprzez znalezienie minimalnej i maksymalnej wartości ilości wystąpień oraz wyliczeniu średniej tych dwóch liczb, wynik tego działania był wartością progową przy binaryzacji.

Po wyliczeniu progu binaryzacji, ilości wystąpień zostały poddane porównaniu z nią. Jeżeli ilość wystąpień była większa od progu, oznaczało to bit "1", w przeciwnym wypadku, było to kodowane jako "0".

Diagram pokazany na rysunku 3.5 przedstawia schemat blokowy algorytmu odczytu danych z czujnika DHT-22.



Rysunek 3.5. Diagram odczytu danych z czujnika DHT-22

## 4. Elektroniczny system pomiarów

Układ pomiarowy składa się z wcześniej przedstawionych urządzeń:

1. mikrokomputer BeagleBone Black
2. czujnik ciśnienia i temperatury BMP085
3. czujnik wilgotności i temperatury DHT-22

Kolejnym krokiem przy tworzeniu systemu pomiarów warunków środowiskowych i meteorologicznych jest poprawne podłączenie czujników do mikrokomputera, aby móc zbierać dane. Wykorzystując tabelę PINów, zainstalowanych w mikrokomputerze BeagleBone Black, podpięto czujniki według poniższej instrukcji:

### **Czujnik BMP085:**

- GND należy podłączyć do pinu P9.1 BeagleBone’a
- VCC do pinu P9.3
- SCL do pinu P9.19
- SDA do pinu P9.20

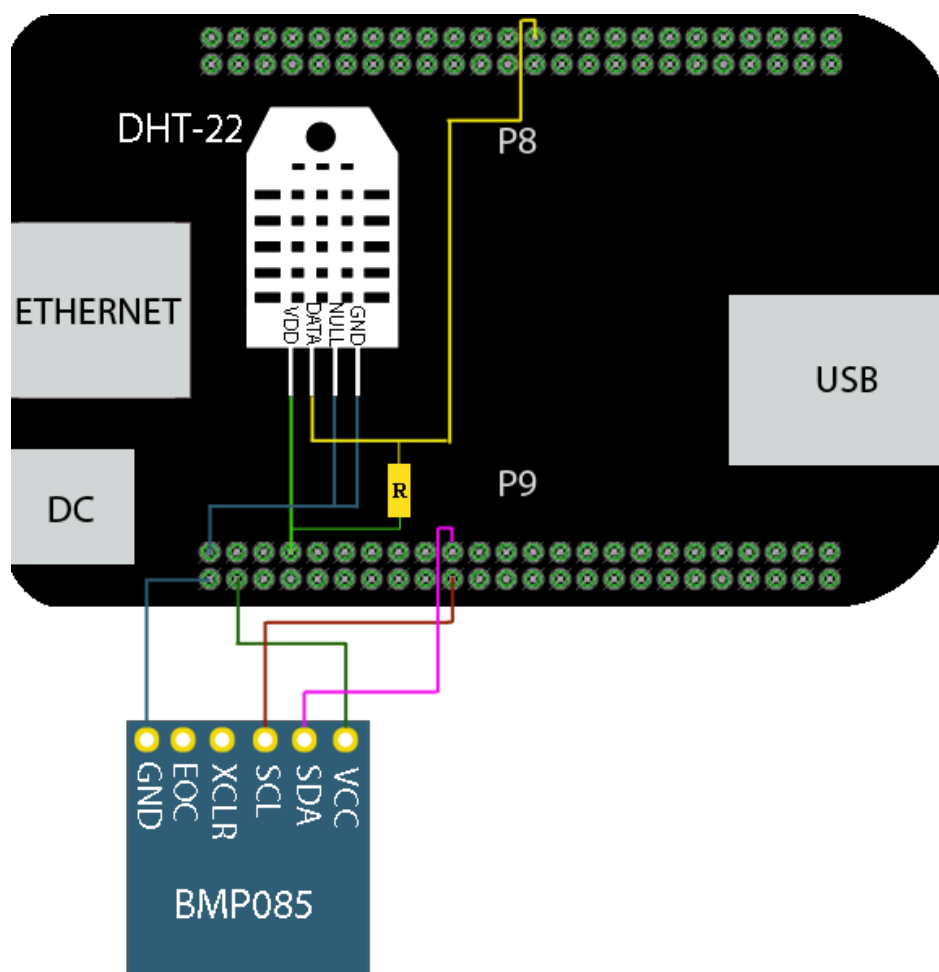
### **Czujnik DHT-22:**

- GND należy podłączyć do pinu P9.2 BeagleBone’a
- VDD do pinu P9.8
- DATA do pinu P8.26

Sposób poprawnego podpięcia układu został również zamieszczony w formie rysunku (rys. 4.1).

Cały układ pomiarowy został połączony ze sobą przy użyciu płytki stykowej. Jest to najlepsze rozwiązanie, jeżeli potrzebne jest tymczasowe rozwiązanie. Takie połączenia można dowolnie i w każdej chwili konfigurować, bez tworzenia nowej płytki drukowanej.

Docelowo planowane jest stworzenie płytki drukowanej w formie nakładki na BeagleBone’a, która idealnie będzie pasować do pinów wyprowadzonych z mikrokomputera.



Rysunek 4.1. Zbudowany układ pomiarowy

Wadami tymczasowego rozwiązania wykorzystującego płytkę stykową jest możliwość zabrudzenia się otworów, co może spowodować wzrost oporu, a nawet brak łączności. Kolejną niedoskonałością jest łatwość przypadkowego rozpięcia układu, które często prowadzi do błędnych pomiarów.

## 5. Komunikacja w układzie

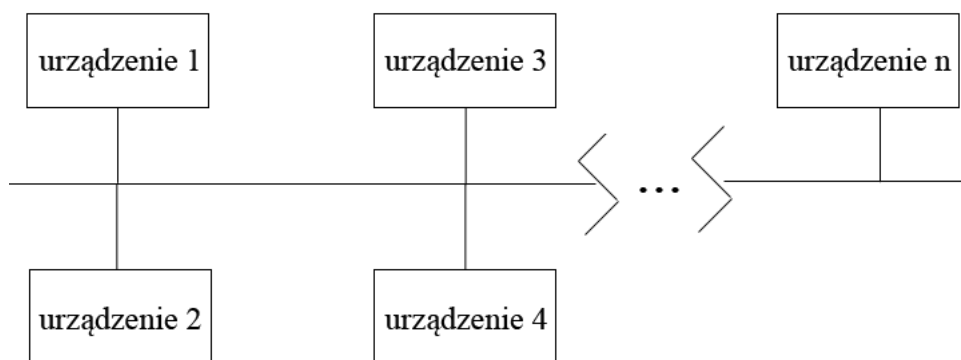
Przesyłanie danych pomiędzy czujnikami a stacją główną (mikrokomputerem) jest najistotniejszą częścią całego układu. System pomiarów, po poprawnym, sprzętowym podłączeniu każdego z podzespołów, należy następnie skomunikować. Urządzenia pomiarowe wraz z głównym komputerem można komunikować poprzez interfejsy cyfrowe zaimplementowane w danym urządzeniu. Mikrokomputer BeagleBone Black obsługuje kilka interfejsów komunikacji: SPI, I<sup>2</sup>C, 1-Wire, CAN, UART. Możliwa jest również łączność poprzez zwykłe porty GPIO (General Purpose Input/Output). W układzie pomiarowym zastosowane zostały interfejsy I<sup>2</sup>C oraz GPIO.

### 5.1. Magistrala szeregową I<sup>2</sup>C

Magistrala jest to układ linii, po których przekazywane są wszystkie informacje pomiędzy podłączonymi do niej urządzeniami, np. komputerem, czujnikiem, regulatorem itp. Zasada działania magistrali opiera się na uzyskiwaniu oraz nadawaniu współpracującym częściom uprawnień do transmisji danych w danej jednostce czasu. W jednej chwili, w magistrali może działać tylko jedno urządzenie nadające oraz dowolna liczba odbiorców. Systemy o budowie opartej na magistrali są łatwo modyfikowalne oraz rozszerzalne, w prosty sposób można dołączyć lub odłączyć elementy systemu. Dane przesyłane na dużą odległość najlepiej jest przekazywać transmisją szeregową, na krótsze odległości, przesyłanie równoległe oraz szeregowe daje podobne rezultaty. Bity oraz całe słowa w tej komunikacji przesyłane są jeden po drugim. Przy takim sposobie łączenia się wystarczą tylko dwa przewody łączące odbiorcę z urządzeniem nadającym.

Przykład urządzenia w magistrali szeregową został przedstawiony na rysunku 5.1.

Jak widać na załączonym schemacie, można podłączyć do magistrali wiele urządzeń. Wszystkie są podłączone do jednej linii danych, na której odbywa się komunikacja. To właśnie przez nią przesyłane są wszystkie dane pomiędzy elementami magistrali.



Rysunek 5.1. Schemat magistrali szeregową

Nazwa magistrala szeregową I<sup>2</sup>C jest akronimem od Inter-Integrated Circuit. Standard został opracowany w latach osiemdziesiątych przez firmę Philips.

Jest ona bardzo często wykorzystywana w układach mikroprocesorowych, w sterownikach wyświetlaczy LCD, można ją stosować do sterowania pamięci RAM, EPROM, układami I/O.

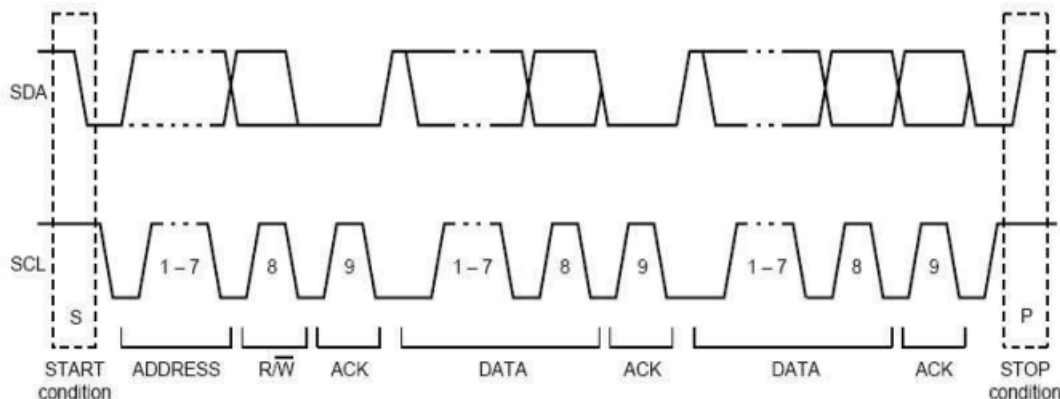
Zaletami magistrali I<sup>2</sup>C są niewątpliwie takie właściwości jak:

- odporność na zakłócenia zewnętrzne
- części układu mogą być dodawane do niego lub wyłączane bez ingerencji w pozostały układ połączeń wcześniej stworzonych
- połączenie na magistrali składają się tylko z dwóch przewodów, przez co ich ogólna liczba jest minimalizowana
- wykrywanie błędów jest proste i łatwe do analizy
- na magistrali może znajdować się wiele urządzeń typu master, umożliwiając kontrolę gotowych układów przez zewnętrzny komputer

Magistrala I<sup>2</sup>C posiada dwie dwukierunkowe linie: dane są przesyłane przez Serial Data (SDA), natomiast sygnał zegara na Serial Clock (SCL).

Rysunek 5.2 prezentuje proces przesyłu danych przez magistralę I<sup>2</sup>C. Gdy nikt nie nadaje, linia zegara - SCL oraz danych - SDA są cały czas w stanie wysokim. W trakcie rozpoczęcia nadawania SCL ma stan wysoki, natomiast linia danych ma zbocze opadające. Po wysłaniu sygnału startu wysyłany jest z każdym taktem zegara bitowo adres urządzenia, z którym nastąpi komunikacja. Po wysłaniu 7 bitów adresu, wysyłany jest bit oznaczający, czy ma nastąpić pisanie czy czytanie z urządzenia slave. Kiedy podrzędna część układu rozpozna, że do niego jest adresowane zapytanie, w takcie dziewiątego bitu wysyła sygnał potwierdzenia przyjęcia dyspozycji. Taka sekwencja wysyłania danych jest powtarzana do momentu pojawienia się sygnału

wysokiego na linii zegara oraz zbocza narastającego na linii danych. Od tego momentu magistrala znowu jest wolna i w dowolnej chwili dowolne urządzenie może zacząć nadawać, aby rozpocząć nową transmisję danych wystarczy powtórzyć powyższą sekwencję kroków.



Rysunek 5.2. Wysyłanie danych poprzez magistralę I<sup>2</sup>C

## 5.2. Komunikacja z wykorzystaniem GPIO

Porty GPIO są binarnymi złączami, które mogą służyć zarówno za wyjście dyskretne, jak i wejście. Ich kierunek jest zazwyczaj konfigurowalny i może być zmieniany w dowolnej chwili. Wejścia/wyjścia są kodowane poprzez odpowiednie napięcie na złączu.

Komunikacja mikrokontrolera oraz urządzeń peryferyjnych, przy użyciu GPIO odbywa się poprzez odpowiednie ustawianie wartości na wyjściu, tak aby drugie urządzenie mogło tą wartość odczytać i przeanalizować. Odpowiedź odbywa się najczęściej przy użyciu tego samego pinu, aby taka funkcjonalność zadziałała musi nastąpić zmiana kierunków pinu w obu częściach układu na przeciwny.



## 6. Programowanie mikrokontrolerów ARM

Procesory ARM posiadają architekturę 32 lub 64 bitową. Z uwagi na energooszczędną konstrukcję używane są przede wszystkim w systemach wbudowanych, w nich niski pobór energii jest jedną z ważniejszych kwestii. Obecnie stały się bardzo popularne i używane są już w niemal każdym urządzeniu mobilnym, np. telefon komórkowy, dyski twarde itp.

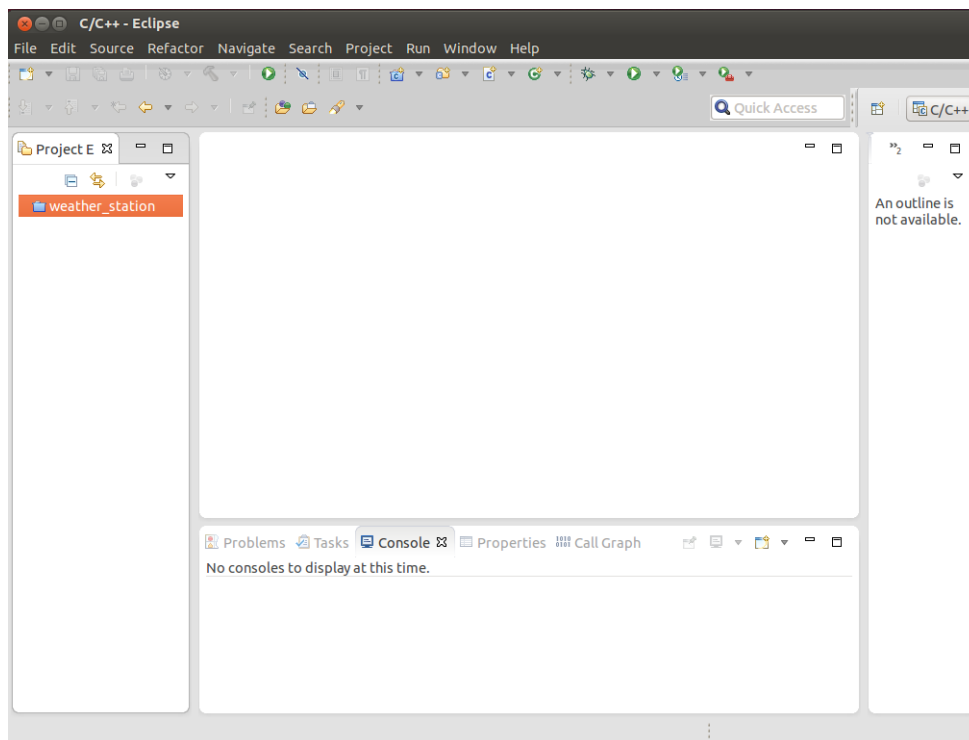
Poniższy rozdział prezentuje w jaki sposób należy kompilować program, aby był wykonalny przez procesor o architekturze ARM oraz jak najwygodniej skonstruować środowisko programistyczne, aby praca z kodem była jak najłatwiejsza.

### 6.1. Środowisko programowania

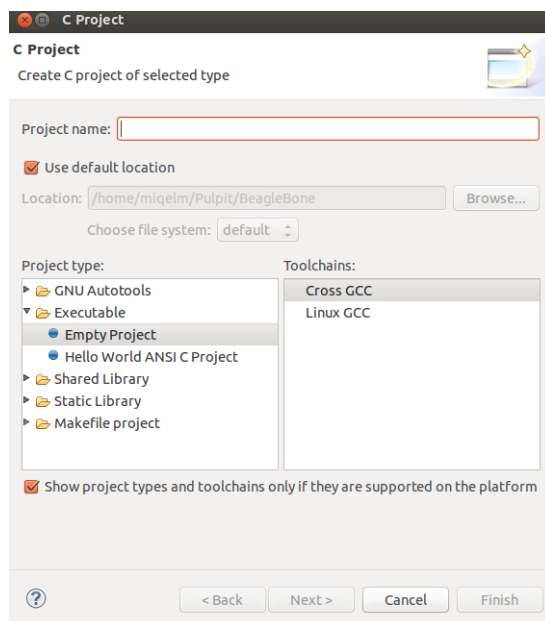
Kod programu oraz obsługi stacji pogody, zaimplementowanej na mikrokontrolerze Beagle-Bone Black, został napisany przy użyciu środowiska programistycznego Eclipse. Środowisko to zostało wybrane przez wzgląd na ogromne możliwości, które ułatwiają w znacznej mierze programowanie oraz skracają jego czas. Eclipse jest darmowym narzędziem do programowania, jest intuicyjny w obsłudze, a przede wszystkim posiada wielką rzeszę użytkowników, przez co w przypadku problemów, ich rozwiązanie jest niemal natychmiastowe. Środowisko to można pobrać z oficjalnej strony, w projekcie został wykorzystany *Eclipse IDE for C/C++ Developers*.

Po rozpakowaniu gotowego środowiska oraz jego uruchomieniu można już zacząć pracę z mikrokontrolerem, wystarczy jeszcze dokonać parę zabiegów, aby czas od zbudowania projektu, do jego uruchomienia na BeagleBone'ie był jak najkrótszy. W celu uzyskania jak najwygodniejszej konfiguracji, Eclipse został uruchomiony na systemie operacyjnym Ubuntu, na którym były przechowywane wszystkie źródła. Dzięki odpowiedniemu dodatkowi do Eclipse'a - Remote System Explorer istnieje możliwość tworzenia programu na komputerze, jego cross-kompilacji do aplikacji wykonywalnej oraz uruchomienia gotowego pliku binarnego na mikrokontrolerze. Dzieje się to dzięki wspomnianemu wcześniej protokołowi SSH.

Aby stworzyć nowy projekt należy kliknąć File->New->C Project (może być również C++), pojawi się okno pokazane na rysunku 6.2:



Rysunek 6.1. Główny wygląd Eclipse'a



Rysunek 6.2. Tworzenie nowego projektu

Po uzupełnieniu nazwy projektu oraz wyboru Cross GCC, należy przejść dalej i zakończyć konfigurację. Teraz następuje najważniejsza rzecz. Aby skompilować projekt i móc go uruchomić na innej platformie sprzętowej, jaką jest procesor ARM, potrzebny jest cross-kompilator. Jest to wymagane, gdyż komputer oraz BeagleBone różnią się budową oraz sposobem komuni-

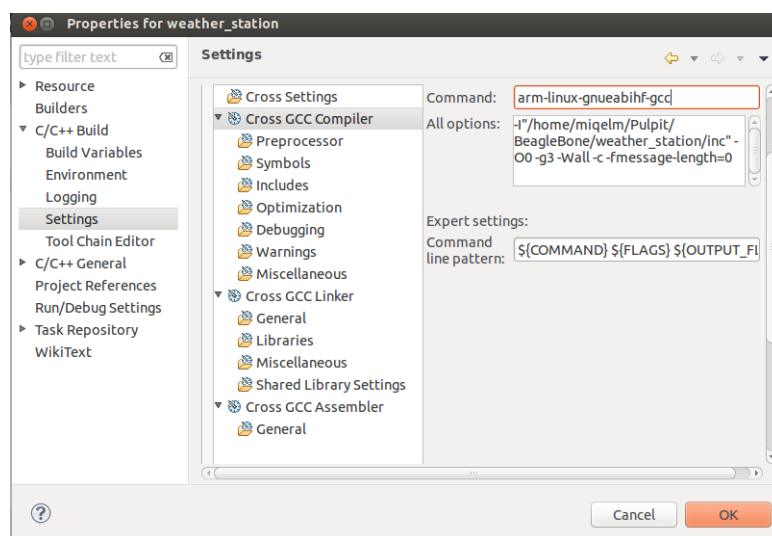
kacji na najniższym poziomie. W związku z tym, należy na komputerze zainstalować narzędzie umożliwiające generowanie pliku binarnego na inną platformę sprzętową, proces ten jest nazywany cross-kompilacją.

BeagleBone Black jest wyposażony w procesor o architekturze ARM hard float, dlatego też potrzebny jest do niego kompilator, nazywa się on *arm-linux-gnueabihf-gcc*. Aby go zainstalować, należy w konsoli użytkownika wpisać następującą komendę:

```
sudo apt-get install arm-linux-gnueabihf-gcc
```

Potwierdzając chęć zainstalowania oraz pomyślnym przebiegu instalacji, można teraz skompilować program na architekturę ARM.

W Eclipse klikając teraz prawym przyciskiem myszy na nowo stworzony projekcie, następnie wciśnięciu Properties, ukazują się właściwości projektu. Należy teraz zakomunikować środowisku, że program będzie kompilowany przy użyciu zainstalowanego przed chwilą kompilatora. W tym celu należy uruchomić zakładkę C/C++ Build, a potem opcję Settings i Cross GCC Compiler, w polu Command należy wpisać nazwę cross-kompilatora. Poniżej zostaje zamieszczony zrzut ekranu przedstawiający zaistniałą sytuację:



Rysunek 6.3. Ustawienia projektu

W podobny sposób należy wypełnić również pola Command w zakładkach Cross GCC Linker oraz Cross GCC Assembler, ten ostatni należy wypełnić wpisując: *arm-linux-gnueabihf-as*.

Tak przygotowany projekt wraz z zainstalowanym cross-kompilatorem umożliwia użytkownikowi bezproblemowe skompilowanie całego projektu do jednego pliku wykonywalnego. Otrzymany plik binarny należy przy użyciu protokołu SSH przesłać do BeagleBone'a i tam go uruchomić.

## 6.2. Używanie bibliotek Linux'a

System Linux w swoim jądrze posiada zaimplementowaną ogromną ilość przydatnych funkcji. Przy tworzeniu systemu pomiarów warunków meteorologicznych użyta została biblioteka funkcji do obsługi magistrali I<sup>2</sup>C.

Funkcje te znacznie ułatwiły odczyt i zapis do czujnika BMP085, który działa tylko i wyłącznie w wymienionym poniżej sposobie komunikacji. Lista funkcji użytych w tworzeniu projektu:

- `i2c_smbus_read_word_data` - służy do odczytywania 16 bitów spod adresu podanego w argumencie
- `i2c_smbus_write_byte_data` - zapis podanego bajtu pod odpowiedni adres w pamięci urządzenia w magistrali
- `i2c_smbus_read_byte_data` - odczyt bajtu z odpowiedniego adresu urządzenia

## 7. Tworzenie aplikacji zbierającej dane pomiarowe

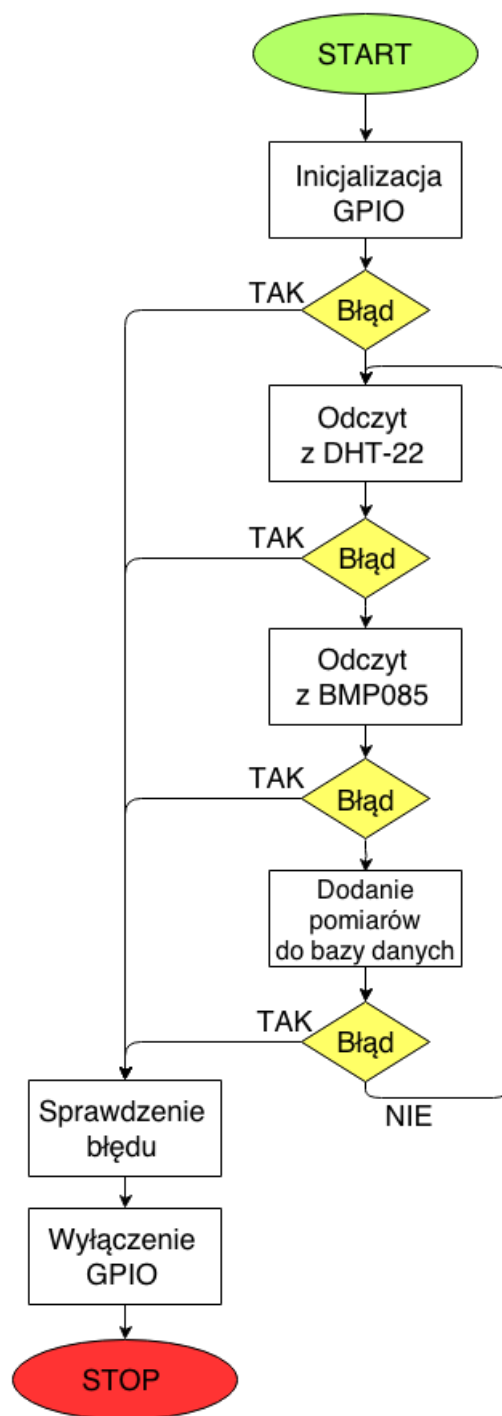
Aplikacja do odbierania danych z czujnika, ich obrabiania oraz przesłania do żądanej bazy danych została całkowicie napisana w języku C. Jest to jeden z języków najczęściej używanych do obsługi systemów wbudowanych. Posiada ogromną ilość bibliotek, które usprawniają pisanie kodu i aplikacji. Dzięki stworzonym bibliotekom można skupić się na tworzeniu programu zamiast tracić czas na tworzenie każdej najmniejszej funkcjonalności od początku.

Główny program stacji pogodowej zawiera w sobie podprocedury do obsługi czujników oraz transmitowaniu zebranych danych do bazy. Odpowiedni podział na podfunkcję sprawia, że kod jest czytelniejszy oraz łatwiej jest go zrozumieć dla osób postronnych. Taka zasada została uwzględniona przy tworzeniu aplikacji do obsługi pomiarów meteorologicznych.

Kwestie bezpieczeństwa również zostały zaimplementowane. W kodzie istnieje kilkanaście różnych wartości oznaczających błędy, jeżeli którykolwiek z nich będzie miał miejsce, nie pozwoli to na dalsze wykonywanie programu. Aplikacja w takim przypadku zostanie zatrzymana, a problem, który nastąpił zostanie zgłoszony obsługującemu w celu jego wyeliminowania.

Kody błędów są zwracane w przypadku, gdy jakaś część programu zostanie wykonana niepoprawnie. Może to być spowodowane chwilowym brakiem łączności z czujnikami lub brakiem dostępu do internetu, aby dodać pomiary do bazy danych. Mogą również wystąpić błędy systemowe, jak np. niepozwolenie na otwarcie pliku. Każdy z problemów może okazać się groźny dla wykonywanej aplikacji, dlatego musi nastąpić zamknięcie całej funkcjonalności.

Na rysunku 7.1 przedstawiony został schemat blokowy głównej funkcji programu stacji pogodowej. Widać na nim, że w przypadku błędu, wszystko zostanie poprawnie zakończone, aby nie spowodować uszkodzenia jakiegokolwiek części układu pomiarowego.



Rysunek 7.1. Diagram programu głównego stacji pogodowej

## Używanie MySQL z poziomu języka C

Baza danych MySQL zostanie szerzej przedstawiona w następnym rozdziale.

Posiadając odebrane prawidłowe pomiary z czujników, należy je następnie zamieścić w bazie danych, aby były one dostępne do wyświetlania lub analizowania. W tym celu potrzebny jest sposób na umieszczenie danych do tej bazy z poziomu programu napisanego w języku C.

Producent bazy danych MySQL udostępnił interfejs komunikacyjny bazy, gotowy do zaimplementowania w C. Nazywa się on MySQL C API i jest dostępny dla wszystkich użytkowników za darmo. MySQL C API jest to biblioteka funkcji, która umożliwia komunikację programu z bazą danych MySQL.

W celu skorzystania z udostępnionej biblioteki najpierw należy ją skompilować do użytku na mikrokomputerze BeagleBone Black. Na stronie głównej projektu MySQL są do ściągnięcia źródła tej biblioteki, po ich pobraniu należy przy pomocy cross-kompilatora skompilować je do biblioteki.

Dodatkowo należy również zainstalować obsługę MySQL na mikrokomputerze, cała procedura instalacji jest ujęta w jednej komendzie, w konsoli należy wpisać:

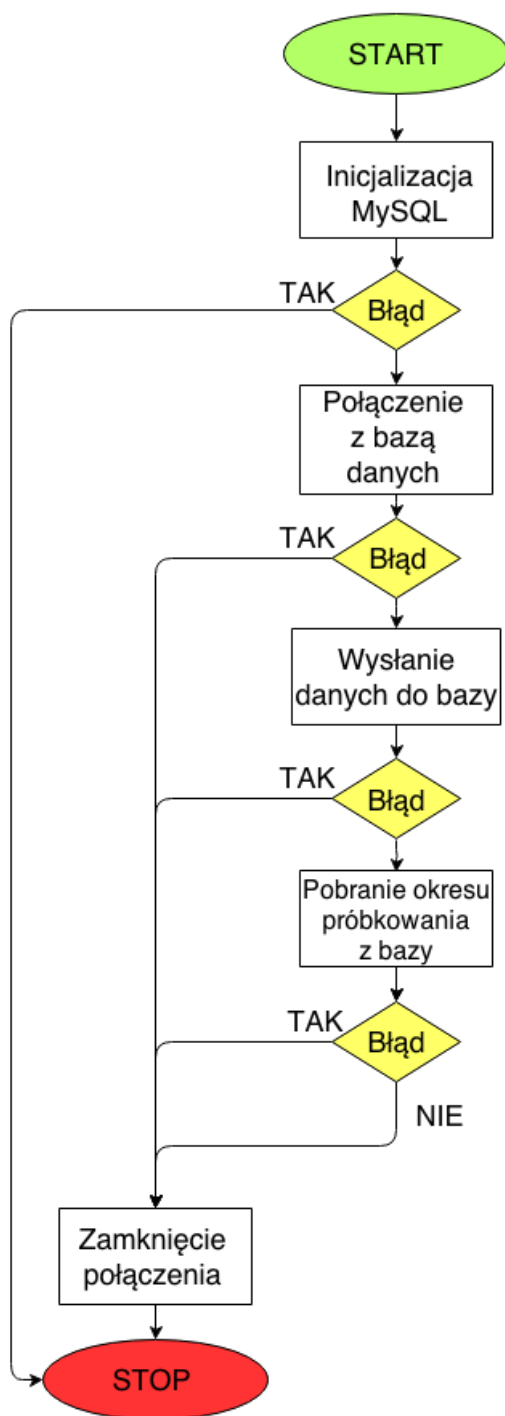
```
sudo apt-get install libmysqlclient-dev
```

Po poprawnym zainstalowaniu i skompilowaniu biblioteki można przejść do kompilacji całego projektu, który będzie łączył się z podaną bazą danych oraz wyciągał z niej i wpisywał do niej rekordy.

Korzystanie z API przygotowanego przez producenta bazy danych jest bardzo proste i wymaga niewielkiej wiedzy początkowej. Dokumentacja jest opisana bardzo jasno oraz można znaleźć dużo poradników na temat używania interfejsu komunikacji z bazą danych.

Diagram przedstawiony na następnej stronie na rysunku 7.2 ukazuje schemat blokowy funkcji łączącej się z bazą danych, wstawianiu danych uzyskanych w nowym pomiarze oraz pobraniu z bazy okresu próbkowania między pomiarami.

Zgodnie z zasadą pisania całego oprogramowania, każdy napotkany błąd jest przekazywany dalej, a w głównej funkcji obsługi stacji pogodowej jest on wyświetlany oraz kończony jest program. Gdy problem nastąpi po nawiązaniu połączenia z bazą, połączenie to musi zostać zakończone, aby nie doszło do wycieków pamięci.



Rysunek 7.2. Diagram programu dodającego dane do bazy MySQL



## 8. Przechowywanie oraz wyświetlanie wyników

Pobrane dane z czujników muszą być przechowane, aby mogły być później odczytane i przeanalizowane. Jest kilka sposobów na gromadzenie danych, najczęściej używanymi jest korzystanie z plików tekstowych lub baz danych. W projekcie inżynierskim wykorzystana została baza danych MySQL.

### 8.1. Baza danych MySQL

MySQL jest systemem bazy danych opartych na strukturalnym języku zapytań SQL. Używa on tego języka do zapisywania oraz pobierania informacji, zarówno z, jak i do bazy. Pomiary ze stworzonej stacji pogodowej zostają wysyłane nieustannie, w trakcie działania programu, do bazy danych MySQL. Użyta w niniejszej pracy baza została założona na serwerze AGH i skonfigurowana poprzez stronę <http://mysql.agh.edu.pl>. Baza przechowuje wartości wszystkich zmierzonych parametrów meteorologicznych od pierwszego uruchomienia stacji pogody, posiada również dane konfiguracyjne - odstęp w czasie pomiędzy badaniem warunków.

Struktura bazy danych jest prosta, posiada ona tabelę o nazwie "weather\_station", w której to właśnie bezpośrednio przechowywane są wartości pomiarowe.

Składa się ona z sześciu kolumn:

1. godzina wykonania pomiaru i wysłania do bazy
2. data
3. temperatura z czujnika DHT-22
4. wilgotność z czujnika DHT-22
5. temperatura z czujnika BMP085
6. ciśnienie z czujnika BMP085

Każda kolumna z pomiarami jest typem float (zmiennoprzecinkowa liczba).

## 8.2. Strona internetowa z wynikami pomiarów

Interfejs użytkownika został stworzony jako strona internetowa w technologii PHP. Jest to język tworzenia dynamicznych stron WWW, nietrudny i wygodny w użytkowaniu. Pozwala również na połączenie z bazą danych i pobieraniem oraz wstawianiem do niej danych.

Strona internetowa, która została stworzona specjalnie do wizualizacji wyników ma bardzo prosty i intuicyjny interfejs użytkownika.

Główną jego częścią są ostatnie wskazania z czujników. Poniżej tych znajdują się wykresy historii pomiarów każdego parametru. Stworzone są dwa wykresy, na pierwszym porównane są ze sobą wskazania temperatur z obu czujników, natomiast na drugim wyświetlana jest wilgotność powietrza oraz ciśnienie atmosferyczne.

Użytkownik posiada, wprost ze strony, możliwość zmiany odstępu czasu jaki ma następować pomiędzy kolejnymi badaniami warunków meteorologicznych. Udostępniona została również funkcjonalność dowolnego ustawiania zakresu dat, dla których mają zostać wyświetlone pomiary w postaci wykresu. Użytkownik sam może decydować, z kiedy chce obserwować pomiary.

Stacja pogody - pomiary

Początkowa data

Godzina:  Minuta:  Sekunda:  Dzień:  Miesiąc:  Rok:

Końcowa data

Godzina:  Minuta:  Sekunda:  Dzień:  Miesiąc:  Rok:

Opóźnienie [s]:

Data	Godzina	DHT-22		BMP085	
		Temperatura	Wilgotność	Temperatura	Ciśnienie
2014-01-07	03:13:24	26.3 °C	31.4%	27.1 °C	988.67 hPa

Rysunek 8.1. Stworzona strona internetowa

Wykresy do pomiarów zostały wykonane przy pomocy biblioteki wykonanej w technologii JavaScript o nazwie Flot.

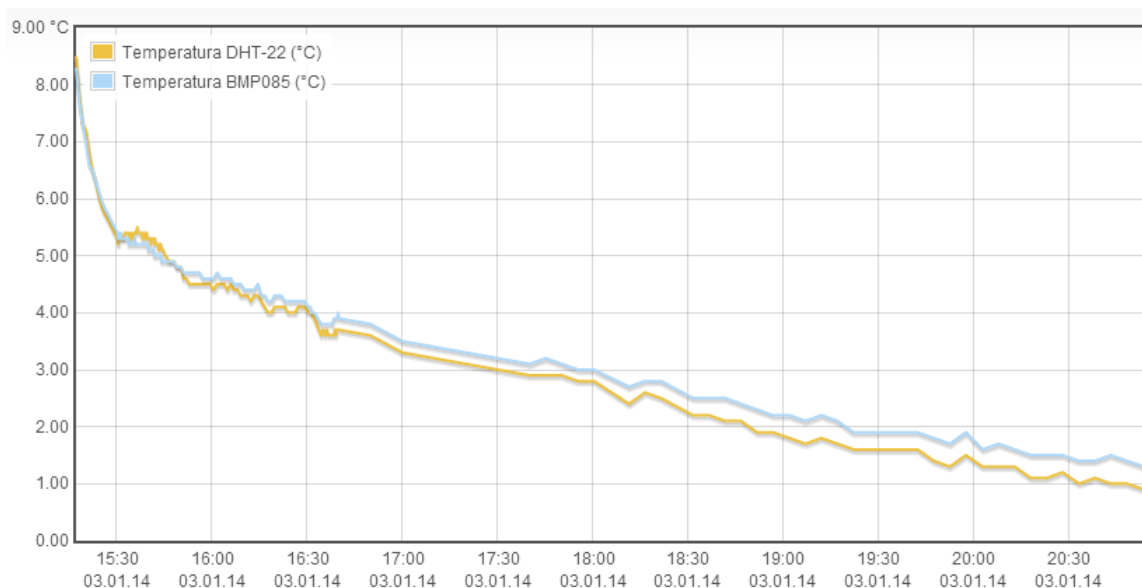
## 9. Rezultaty pomiarów

Testy całego systemu pomiarowego zostały przeprowadzone w dwóch różnych środowiskach. W pierwszym przypadku mierzone były warunki atmosferyczne panujące na zewnątrz budynku mieszkalnego. Drugi test został przeprowadzony w warunkach mieszkalnych.

### 9.1. Warunki zewnętrzne

Cały układ pomiarowy został testowo wykorzystany do badania warunków meteorologicznych panujących na wolnym powietrzu. Aplikacja zbierająca dane została uruchomiona na kilka godzin (od około 15:00 do 21:00).

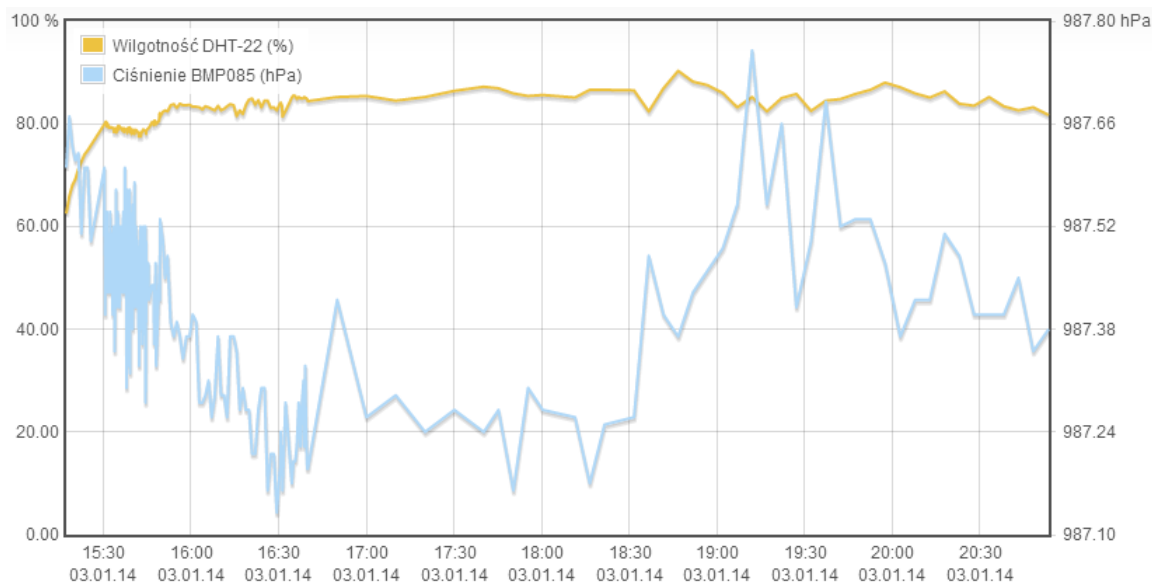
Poniższe wykresy prezentują warunki wtedy panujące. Jak widać na załączonych rysunkach, temperatura stale maleje wraz z upływem czasu. Świadczy to o poprawnym pomiarze, gdyż wraz z zachodzącym słońcem, robi się coraz chłodniej.



Rysunek 9.1. Pomiar temperatury na zewnątrz budynku

Na wykresie 9.1 można zaobserwować stałą różnicę pomiędzy wskazaniami temperatur pomiędzy czujnikami DHT-22 oraz BMP085. Różnice w ich pomiarach wynoszą około 0,5 do 1 stopnia celsjusza i są spowodowane różnymi dokładnościami urządzeń mierzących.

Rysunek 9.2 przedstawia zależności wilgotności powietrza oraz ciśnienia atmosferycznego od czasu, w którym następowało badanie.

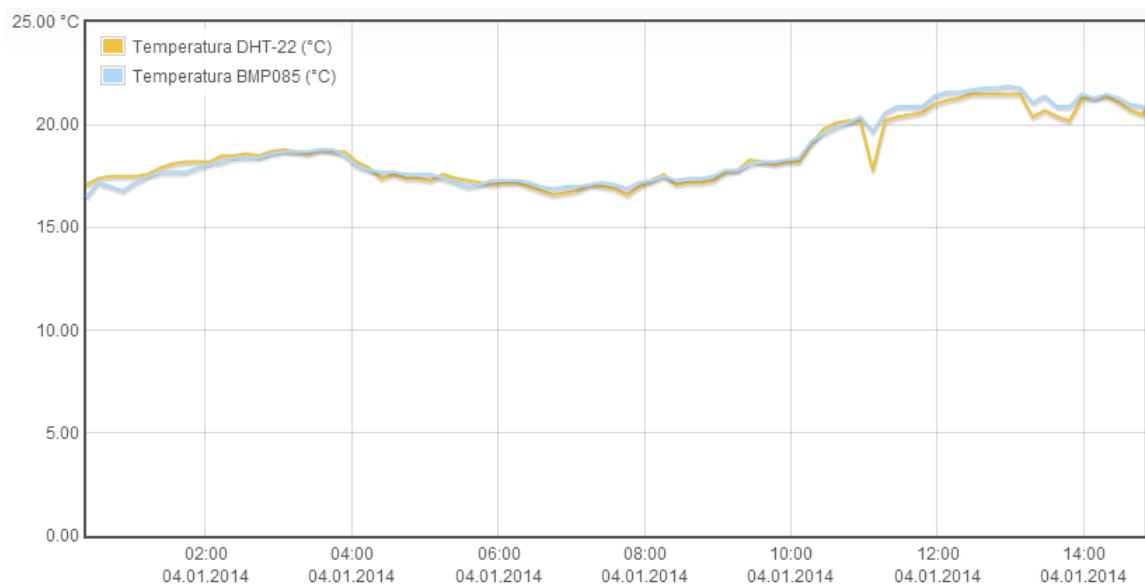


Rysunek 9.2. Pomiar wilgotności oraz ciśnienia na zewnątrz budynku

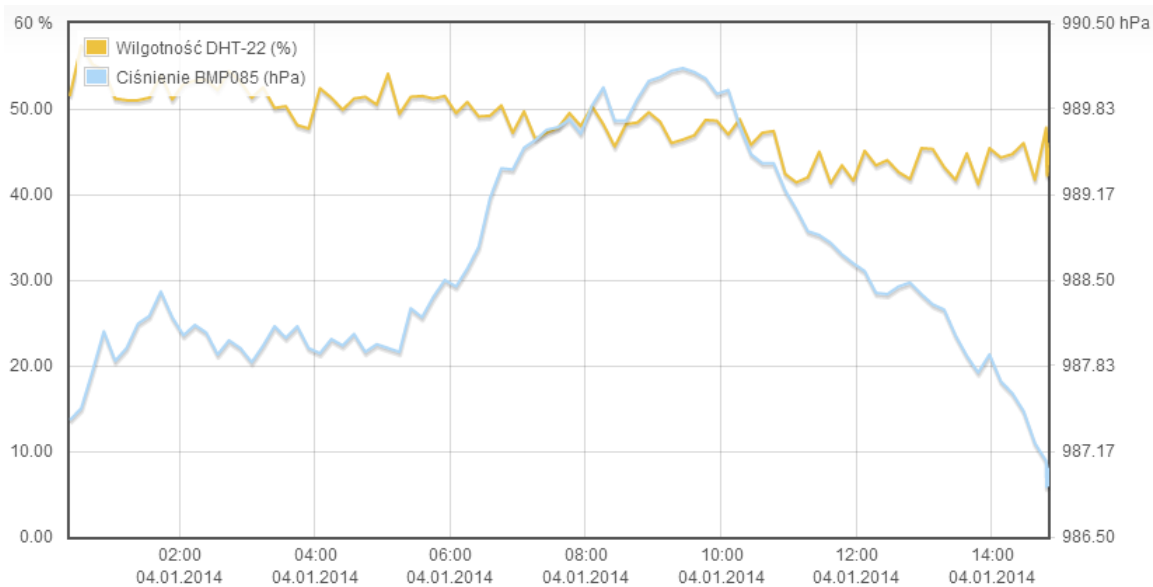
Wahania ciśnienia atmosferycznego widoczne na wykresie 9.1 spowodowane są szumem występującym na czujniku oraz jego dokładnością.

## 9.2. Warunki pokojowe

Następnie stacja pogodowa została ustawiona w pokoju, gdzie temperatura nie ulega dużym wahaniom i uruchomiono aplikację zbierającą pomiary. Wykres na rysunku 9.3 przedstawia zależność temperatur od czasu, natomiast rysunek 9.4 prezentuje pomiary ciśnienia atmosferycznego oraz wilgotności. Jak można zauważyć z wykresu temperatur, w nocy miała ona stosunkowo niską wartość, natomiast nad ranem zaczęła rosnąć. Wiąże się to z faktem otwarcia okna oraz chłodzenia pomieszczenia, w celu uzyskania lepszego wypoczynku podczas snu oraz zamknięcia okna nad ranem.



Rysunek 9.3. Pomiar temperatury wewnątrz pomieszczenia



Rysunek 9.4. Pomiar wilgotności oraz ciśnienia wewnątrz pomieszczenia

## 10. Podsumowanie

Projekt inżynierski opisany w niniejszej pracy ma na celu badanie warunków meteorologicznych badając trzy pomiary: temperaturę, ciśnienie oraz wilgotność powietrza.

Testy przeprowadzone po ukończeniu tworzenia układu oraz oprogramowania dowodzą, że projekt inżynierski został zakończony sukcesem, wszystkie zaplanowane funkcjonalności zostały w 100 % zrealizowane. Pomiary warunków meteorologiczne, które zostały zmierzone były bardzo zbliżone do wskazań otrzymanych przy pomocy standardowych mierników, jak np. termometr pokojowy.

Największym problemem napotkanym podczas tworzenia niniejszego projektu było doprowadzenie do kompilacji pełnego projektu, wraz ze wszystkimi bibliotekami, cross-kompilowaniem oraz uruchamianiem aplikacji na innej architekturze niż na tej, na której tworzone było oprogramowanie.

Możliwymi koncepcjami rozwinięcia tematyki niniejszej pracy dyplomowej jest stworzenie i uruchomienie algorytmów do przewidywania pogody. Jest to zagadnienie bardzo trudne i skomplikowane, jednakże na pewno bardzo interesujące oraz pochłaniające. Numeryczne obliczanie prognozy pogody może bardzo ułatwić użytkownikowi życie.

Inną możliwością jest zastosowanie mikrokomputera BeagleBone oraz czujnika ciśnienia atmosferycznego do zbudowania urządzenia latającego, np. kwadrokoptera.

## Bibliografia

- [1] Jacek Bogusz *Lokalne interfejsy szeregowo w systemach cyfrowych*. Wydawnictwo BTC, Warszawa, 2004.
- [2] Wojciech Mielczarek *Szeregowo interfejsy cyfrowe*. Wydawnictwo HELION, Gliwice, 1993.
- [3] Michael Leonard <http://www.michaelhleonard.com/cross-compile-for-beaglebone-black/>. [Dostęp: 11.12.2013].
- [4] BeagleBone Black Wiki <http://circuitco.com/support/index.php?title=BeagleBoneBlack>. [Dostęp: 11.12.2013].
- [5] Strona projektu ARM Hard Float <http://www.armhf.com/>. [Dostęp: 11.12.2013].
- [6] Specyfikacja czujnika ciśnienia BMP085 <https://www.sparkfun.com/datasheets/Components/General/BMP085-DS000-05.pdf>. [Dostęp: 11.12.2013].
- [7] Specyfikacja czujnika ciśnienia DHT-22 <http://www.adafruit.com/datasheets/DHT22.pdf>. [Dostęp: 11.12.2013].
- [8] Luke Welling, Laura Thomson *PHP i MySQL. Tworzenie stron WWW. Vademecum dla profesjonalisty*. Wydanie czwarte.. Wydawnictwo HELION, Gliwice, 2009.
- [9] Strona główna projektu FlotChart <http://www.flotcharts.org/>. [Dostęp: 6.01.2014].
- [10] Strona główna środowiska programistycznego Eclipse <http://www.eclipse.org/>. [Dostęp: 6.01.2014].
- [11] Poradnik korzystania z MySQL C API <http://zetcode.com/db/mysqlc/>. [Dostęp: 6.01.2014].