Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Meteorologická stanice na platformě Arduino

Jan Gemperle

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jan Gemperle

Informatika

Název práce

Meteorologická stanice na platformě Arduino

Název anglicky

Weather station based on Arduino platform

Cíle práce

Hlavním cílem této bakalářské práce je vytvoření funkční meteorologické stanice pomocí mikrokontroléru Arduino. Dalším cílem pak bude vytvoření bezdrátového čidla, které bude tvořit několik senzorů a bude poskytovat hodnoty, které se zobrazí na displeji.

Metodika

V této bakalářské práci budou popsány základní informace o meteorologických stanicích, jejich rozdělení a využívaná čidla. V další části budou zobrazeny základní vlastnosti mikrokontroléru Arduino spolu s vhodnými senzory pro tvorbu meteostanice. Dále pak bude popsána samotná konstrukce meteorologické stanice s vlastním schématem zapojení a jednotlivými částmi kódu. Následně bude provedeno měření v reálných podmínkách. Naměřená data budou zpracována do uživatelsky přehledného formátu, který bude zobrazen na displeji meteostanice. Veškerý postup bude popsán a zhodnocen.

Doporučený rozsah práce

30-40 stran

Klíčová slova

meteorologická stanice, meteostanice, Arduino, senzory

Doporučené zdroje informací

Banzi, M.– Getting Started with Arduino (2nd Edition), Maker Media, Inc., 2011, ISBN-10: 1449309879

BELL, C A. *Beginning sensor networks with Arduino and Raspberry Pi.* [New York, New York]: Apress, 2013. ISBN 1430258241.

McRoberts, M.– Beginning Arduino: Technologyin Action (2nd Edition), Apress,2013, ISBN-10: 143025016X

MONK, S.–Programming Arduino Getting Started with Sketches (1st Edition), McGraw-Hill Education, 2011, ISBN-10:0071784225

Předběžný termín obhajoby

2018/19 LS – PEF

Vedoucí práce

Ing. Marek Pícka, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 24. 1. 2019

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 24. 1. 2019

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 05. 03. 2019

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Meteorologická stanice na platformě Arduino" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 5.3.2019 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Marku Píckovi, Ph.D. za přínosné konzultace a vedení mé práce a také mému tátovi Ing. Jiřímu Gemperlemu za rady a sestavení konstrukce pro venkovní čidlo.

Meteorologická stanice na platformě Arduino

Abstrakt

Tato bakalářská práce se zabývá navržením, sestavením a testováním meteorologické stanice na platformě Arduino. Meteostanice se skládá ze dvou samostatných čidel využívající mikrokontroléry Arduino, které mezi sebou bezdrátově komunikují. Venkovní čidlo je využito pro získávání venkovních hodnot a jejich odesílání. Vnitřní čidlo sbírá vnitřní hodnoty a přijímá hodnoty z venkovního čidla. Následně všechny hodnoty zobrazuje na displeji.

V teoretické části jsou uvedena základní fakta o meteorologických stanicích, jejich rozdělení a nejčastěji měřené meteorologické jevy. Poté následuje návrh vlastní meteostanice s výběrem vhodných senzorů a jejich schématickým zapojením.

Praktická část se zabývá sestavením jednotlivých senzorů do navrhovaných čidel, vytvořením programu a následným testováním.

Klíčová slova: Arduino, Bezdrátové čidlo, Meteorologická stanice, Meteostanice, Měření atmosférických jevů, Senzory

Weather station based on Arduino platform

Abstract

This bachelor thesis deals with the design, construction and testing of the weather station based on the Arduino platform. The weather station consists of two separate units using Arduino microcontrollers that communicate wirelessly with each other. The outdoor unit is used to retrieve outdoor values and send them. The indoor unit collects internal values and receives values from the outdoor unit. Then all values are illustrated on the display.

In the first part are presented basic facts about weather stations, their distribution and the most frequently measured meteorological phenomena. Then follows the design of their own weather station with a selection of suitable sensors and their schematic connection.

The second part deals with assembly of individual sensors into the proposed units, creation of a program and subsequent testing.

Keywords: Arduino, Atmospheric Measurement, Meteo Station, Weather Station, Wireless Sensor, Sensors

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

I^2C	Inter Integrated Circuit
IDE	Integrated development environment
LCD	Liquid Crystal Display
LED	Light Emitting Diode
RH	Relative Humidity
SPI	Serial Peripheral Interface
TFT	Thin Film Transistor
USB	Universal Serial Bus

1 Úvod

V dnešní době si už chod společnosti bez meteorologických stanic nedokážeme představit. Využívají se v nepřeberném množství odvětví, které mají alespoň něco společného s měřením atmosférických jevů. Ať už se jedná o profesionální předpověď počasí, inteligentní vytápění budov nebo jen měření venkovní teploty teploměrem za oknem. Všechny tyto činnosti mají jedno společné: získávají hodnoty podle již dříve zavedených a odzkoušených postupů, a daná měření tak získávají smysl pro širokou veřejnost.

Meteostanice v této práci se skládá ze dvou čidel, která mají samostatné napájení a bezdrátově spolu komunikují. Přesněji venkovní čidlo odesílá naměřené hodnoty vnitřnímu čidlu, které hodnoty zpracovává a zobrazuje na displeji tak, aby byly dobře čitelné a srozumitelné. Každé čidlo má svůj vlastní mikrokontrolér Arduino, na kterou jsou připojeny zvolené senzory, které umožňují dané měření. Postupy pro získání měřených hodnot, jsou již ověřeny a řádně otestovány. Hodnoty jsou uváděny v jednotkách, které využívá široká veřejnost.

Bakalářská práce je rozdělena do čtyř částí. První část popisuje cíl a způsob zpracování práce. Jaká stanice má být vytvořena a jak jsou vybrané hodnoty zobrazovány a vyhodnocovány.

V druhé části jsou zobrazeny stručné informace o meteorologických stanicích a jejich rozdělení. Poté následuje výpis vhodných vývojových desek a komponentů ke zhotovení meteorologické stanice spolu s jejich specifikacemi.

Třetí část popisuje samotné zhotovení stanice. Je zde také vysvětlen program po jednotlivých částech. Dále popisuje návrh plošných spojů, které byly pro tuto meteorologickou stanici navrženy.

Vyhodnocení výsledků se nachází ve čtvrté části. Získaná data jsou porovnána s daty jiné certifikované meteostanice.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem této bakalářské práce je vytvoření funkční meteorologické stanice pro domácí využití pomocí mikrokontroléru Arduino. Meteorologická stanic bude tvořena spolehlivými a přesnými senzory, které budou dobře dostupné a budou mít přijatelnou cenu.

Dalším cílem je využít bezdrátovou technologii pro komunikaci mezi venkovním a vnitřním čidlem, díky které nemusejí být obě čidla propojena kabelem, a tím zjednodušit instalaci pro reálné testování.

Posledním cílem je porovnat naměřené hodnoty s certifikovanými hodnotami z jiných meteorologických stanic a určit, do jaké míry jsou data použitelná. Toto porovnání se bude především vztahovat na hodnoty měřené venkovním čidlem.

2.2 Metodika

Tato bakalářská práce je vypracována na základě dostupných internetových zdrojů, datasheetů ke komponentům a odborné literatury.

V teoretické části jsou popsány základní informace o meteorologických stanicích, doplněné informativními obrázky. Dále je zde vyobrazeno základní rozdělení meteostanic spolu se stručnou historií o počátku zaznamenávání údajů o počasí v České republice. Jako poslední jsou v této části uvedeny informace o použitých mikrokontrolérech Arduino spolu s vhodnými komponenty pro vytvoření vlastní meteorologické stanici.

V praktické části této práce je vyobrazen návrh meteorologické stanice spolu s návrhem plošných spojů pro venkovní i vnitřní čidlo, které mají usnadnit připojení komponentů k mikrokontrolérům. Dále je v této kapitole popsán a vysvětlen program, který umožňuje čidlům měřit a zaznamenávat hodnoty ze senzorů. Také je v této části zaznamenána kalibrace senzorů, díky které by naměřené hodnoty měly být přesnější a lépe odpovídat skutečnosti.

V poslední části jsou vyhodnoceny výsledky spolu s porovnáním hodnot měřených na této meteorologické stanici a hodnot měřených certifikovanou meteostanicí.

3 Teoretická východiska

3.1 Meteorologické stanice

Pod pojmem meteorologická stanice si můžeme představit definici: „*Místo, v němž se konají stanovená meteorologická pozorování podle dohodnutých mezinárodních nebo vnitrostátních postupů*.“ Jedná se tedy o místo, kde se měří námi zvolené hodnoty v určitém čase. Tyto hodnoty se pak kódují do předem dohodnutých zpráv.

Meteorologické měření spočívá v měření teploty vzduchu, vlhkosti vzduchu, atmosférického tlaku, délky a doby slunečního svitu, výšky srážek, výšky sněhové pokrývky, rychlosti a směru větru, dohlednosti, sledování oblačnosti, bouřek či dalších jevů, které mohou být různé podle specializace. K měření těchto jevů a prvků slouží specializované senzory. (In-počasí, 2017a)

3.1.1 Historie meteorologických stanic

Nejstarší dochované záznamy z přístrojově měřené teploty na našem území se datují na počátek 18. století. Mezi nejstarší meteorologické stanice patří hvězdárna Klementinum (Obrázek 1), která se pyšní nejdelším souvislým měřením na našem území. Tato měření byla zahájena v roce 1752 a pokračují až dodnes. (In-počasí, 2017b)

Obrázek 1- Klementinum. *Zdroj: https://prazskeprochazky.cz/*

3.1.2 Dělení meteorologických stanic

Každá meteorologická stanice může plnit úkoly různého odborného zaměření a rozsahu. (In-počasí, 2017a)

Meteorologické stanice se dají dělit podle zaměření na:

- synoptické stanice,
- klimatologické stanice,
- letecké meteorologické stanice,
- agrometeorologické stanice.

Dále je možné dělení podle umístění na:

- pozemní stanice,
- námořní stanice,
- stanice umístěné na letounech.

Poslední dělení je pak možné podle charakteru získávání dat:

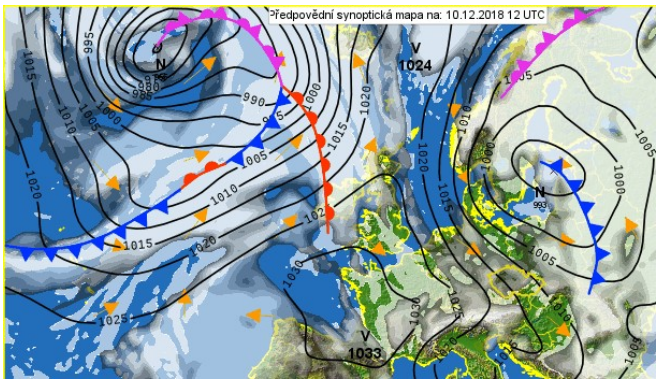
- přízemní stanice,
- aerologické stanice.

3.1.2.1 Dělení podle zaměření

Synoptické stanice

Synoptické stanice mají největší frekvenci získávání měření meteorologických prvků, a to nepřetržitě jednou za hodinu. Data se následně odesílají do meteorologických center, kde se z nich vytvářejí synoptické mapy (Obrázek 2) aktuálního počasí pro nadcházející dny. Mezi měřené prvky patří teplota vzduchu, vlhkost vzduchu, teplota rosného bodu, tlak vzduchu, směr a rychlost větru, teplota půdy, stav počasí, průběh počasí, množství oblačnosti, druhy vyskytujících se oblaků, výška základny oblaků, množství a druh srážek a výskyt všech meteorologických jevů. (Meteorologická stanice Maruška, 2006)

Obrázek 2- Synoptická mapa. *Zdroj: http://portal.chmi.cz/*



Klimatologické stanice

Klimatologické stanice nezaznamenávají měření tak často jako synoptické stanice. Hodnoty se zaznamenávají třikrát denně, a to v 07, 14, 21 hodin SEČ (Středoevropský čas). Měří se meteorologické jevy, jejich druh, intenzita a doba trvání. Mezi měřené prvky může také patřit množství spadlých srážek a výšku napadaného sněhu. Stanic se dají ještě dělit na manuální (Obrázek 3), nebo automatické podle druhu obsluhy. (Meteorologická stanice Maruška, 2006)

Obrázek 3- Klimatologická stanice. *Zdroj: http://maruska.ordoz.com/*

Letecké meteorologické stanice

Letecké stanice slouží zejména k získávání informací pro piloty a letovou dopravu. Umístěny jsou na letištích.

Agrometeorologické stanice

Agrometeorologické (zemědělsko-meteorologické) stanice slouží zejména pro zemědělce, kterým měří například teplotu půdy v různých hloubkách.

3.1.2.2 Dělení podle umístění stanice

Meteorologické stanice se také rozlišují podle polohy umístění. Nejrozšířenější jsou pozemní stanice. Další druh jsou stanice námořní, ty bývají umístěny na bójích, nebo přímo na lodích. Posledním druhem jsou meteostanice umístěné na letounech. (In-počasí, 2017a)

3.1.2.3 Dělení podle charakteru získávání dat

Mezi přízemní meteostanice patří stanice, které provádí měření v přízemní vrstvě atmosféry. Aerologické meteorologické stanice získávají hodnoty z míst, která k přízemní vrstvě už nepřiléhají, tudíž pozemními stanicemi nejsou tato místa dostupná. Pro aerologická měření se nejčastěji používají radiosondy. (http://www.pocasi.astronomie.cz/])

Radiosondy

Jedná se o meteorologické stanice, které jsou přidělané na vodíkový balón, který meteostanici vynese až do výšky 20 km, poté balón praskne. Během své cesty vzhůru radiosonda (Obrázek 4) dokáže změřit aktuální teplotu, vlhkost i tlak. Díky neustálému zaměřování je také možné měřit rychlost a směr větru. (Počasícz.cz)

Obrázek 4- Radiosonda. *Zdroj: https://ok9dmr.mypage.cz/*

3.1.3 Funkce meteorologických stanic

Meteorologická stanice se skládá minimálně ze dvou částí: senzorické a výpočetní části. Senzorická část vlivem okolního prostředí mění své vlastnosti tak, aby výpočetní část

dokázala rozpoznat rozdíl a změřit vybranou veličinu. Počet a typ senzorů závisí na způsobu použití stanice.

3.1.3.1 Měření teploty

Teplota se nejčastěji měří pomocí polovodiče, který při změně teploty změní svoji vodivost. Výpočetní část dokáže tento úbytek napětí zaznamenat a na jeho základě dopočítat okolní teplotu.

3.1.3.2 Měření vlhkosti vzduchu

Relativní vlhkost vzduchu je možné změřit více způsoby, například měření pomocí kapacitního hygrometru. Kapacitní hygrometr je kondenzátor s dielektrikem z polymeru, které mění své vlastnosti při změně vlhkosti. (https://www.vetrani.tzb-info.cz/)

3.1.3.3 Měření tlaku

Senzor tlaku využívá průhybu membrány, díky kterému se mění napětí. Z velikosti napětí lze poté odečíst atmosférický tlak.

3.1.3.4 Měření slunečního svitu

Z měření slunečního svitu získáváme nejčastěji dva údaje, délku a dobu slunečního svitu. Délka slunečního svitu je čas, kdy slunce svítilo na obloze alespoň s intenzitou 120 W/m^2. Doba slunečního svitu je časový interval od východu slunce až do jeho západu. (In-počasí, 2017c)

3.1.3.5 Měření srážek

Přístroj na měření srážek využívá nádoby s nálevkou, ve které měří výšku hladiny pomocí plováku. Množství srážek se udává v milimetrech (1 mm = 1 l/m^2). Pro správné měření sněhu nebo krup je nutné nechat pevnou látku roztát.

3.1.3.6 Měření rychlosti a směru větru

Pro rychlost větru se používají anemometry (Obrázek 5). Vítr otáčí lopatkami přístroje, a tím mění polohu spínače. Rychlost se tedy spočítá z počtu impulzů za měřený

čas. Směr větru se určuje podle velikosti napětí, které se mění spolu s natočením střelky díky rezistorům zabudovaným ve střelce.

Obrázek 5- Senzor pro měření rychlosti a směru větru. *Zdroj: http://www.apu.cz/*

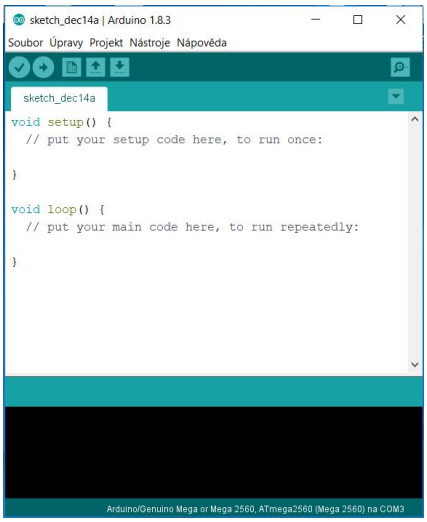
3.2 Mikrokontroléry Arduino

Arduino je open-source hardware umožňující ovládání nepřeberného množství elektronických součástek a zařízení. Jeho první představení se uskutečnilo v roce 2005 s cílem zpřístupnit elektrotechnický hardware a software laické veřejnosti. V dnešní době se mikrokontroléry Arduino prodávají po celém světě. Na internetu existují návody téměř na vše. Důvod je jednoduchý, Arduino je open-source, tudíž jeho schéma zapojení je volně ke stažení, a tak si ho může doma vytvořit skoro každý. Z toho důvodu se po světě začaly prodávat klony. Jedná se o levnější verze originálu, u kterých sice nejsou použity nejkvalitnější součástky, ale jejich funkčnost je dostačující. Tím se mikrokontroléry Arduino dostaly mezi širokou veřejnost a nic už nestálo v cestě vytvoření vlastní komunity.

Mikrokontroléry Arduino se programují pomocí programovacího jazyka Wiring, který je založený na programovacím jazyku C. Vývojové prostředí pro psaní „*sketchů*“ (název kódu), se jmenuje Arduino IDE (Obrázek 6). IDE běží na počítači, ke kterému je mikrokontrolér Arduino připojen pomocí USB, a umožňuje kód zkompilovat a následně nahrát do mikrokontroléru.

Každý *sketch* se skládá ze dvou hlavních metod. Funkce *setup(),* ve které probíhá nastavení vstupů/výstupů a inicializace připojeného hardware. Druhá metoda se jmenuje *loop()* a slouží pro běh celého programového cyklu. Jak název napovídá, jedná se o smyčku, která se neustále opakuje, dokud zařízení neodpojíme od zdroje energie. (Bell, 2013)

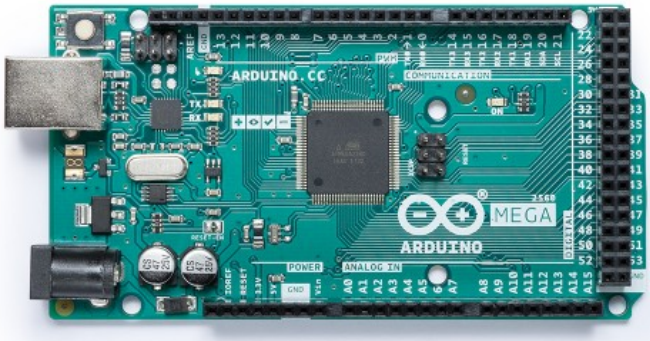
Obrázek 6- Arduino IDE. *Zdroj: vlastní zpracování*



3.2.1 Model Arduino Mega 2560

Model Arduino Mega 2560 (Obrázek 7) je založen na čipu Atmel ATmega2560. Na vývojové desce je možné nalézt 54 digitálních pinů, 16 analogových pinů, USB připojení, reset tlačítko a mnohé další. (Arduino)

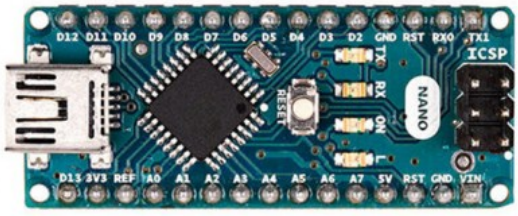
Obrázek 7- Arduino Mega 2560. *Zdroj: https://store.arduino.cc/*



3.2.2 Model Arduino Nano

Model Arduino Nano (Obrázek 8) je založen na kontroléru Atmel ATmega328. Na vývojové desce je 14 digitálních vstupních/výstupních pinů, 8 analogových vstupních pinů, USB připojení a reset tlačítko. (Arduino)

Obrázek 8- Arduino Nano. *Zdroj: https://store.arduino.cc/*



3.3 Komponenty

Digitální a analogové piny dělají z desek Arduino ideální platformu pro připojení různých komponentů. Velkému množství senzorů, modulů, apod. stačí pro správnou funkci pouze jejich připojení a využití správné knihovny v programu. Díky tomu je možné různé druhy komponentů kombinovat a vytvářet tak unikátní sestavy. (Bell, 2013)

3.3.1 Senzory

Senzor je obecný termín pro zařízení, které získává informace řídicímu systému. Pro výběr vhodných senzorů je důležité určit, jaké atmosférické jevy budou měřeny. Z naměřených hodnot je možné získat další hodnoty, například zdánlivou teplotu a rosný bod.

Zdánlivá teplota

Termín zdánlivá teplota je používán od roku 1970, kdy byl navržen vzorec pro její výpočet ve vnitřních prostorách při známé teplotě a relativní vlhkosti. V osmdesátých letech byl vzorec rozšířen o vliv rychlosti větru.

Zdánlivá teplota udává působení vnějších vlivů na oblečeného člověka. Pokud je relativní vlhkost vyšší než její referenční hodnota při určité teplotě, pak je zdánlivá teplota vyšší než právě určená teplota. Toto tvrzení platí i naopak, s nižší relativní vlhkostí klesá i zdánlivá teplota. Dále zdánlivou teplotu ovlivňuje rychlost větru. S přibývající rychlostí zdánlivá teplota lineárně klesá.

Výpočet:

$$AT = Ta + 0{,}33 \cdot E - 0{,}70 \cdot ws - 4{,}00$$

$$E = \frac{rh}{100} \cdot 6{,}105 \cdot e^{17{,}27 \cdot \frac{Ta}{237{,}7+Ta}}$$

Kde AT je zdánlivá teplota, Ta je naměřená teplota, ws je naměřená rychlost větru, E značí tlak vodních par a rh relativní vlhkost. (Thermal Comfort observations)

Rosný bod

Rosný bod je označován jako teplota, při které je okolní vzduch nasycený vodní párou a již není schopen žádnou další pojmout. Pokud teplota klesne pod tento bod, tak se přebytečná vodní pára začne srážet a tvořit kapky.

Spolu s teplotou povrchu je rosný bod důležitým údajem pro předpovídání vzniku mlhy a rosy. Tato předpověď je velice důležitá zejména v zimních měsících, kdy se venkovní teplota pohybuje pod bodem mrazu a vytváří námrazu a ledovku na silnicích.

Výpočet:

$$T_{dp} = \frac{243{,}5 \cdot \ln\left(\frac{V}{100} \cdot e^{\frac{17{,}67 \cdot T}{243{,}5+T}}\right)}{17{,}67 - \ln\left(\frac{V}{100} \cdot e^{\frac{17{,}67 \cdot T}{243{,}5+T}}\right)}$$

Kde T_{dp} je teplota rosného bodu, V je naměřená relativní vlhkost a T je naměřená teplota. (In-počasí, 2017d)

Senzor BME280

BME280 (Obrázek 9) je snímač atmosférického tlaku. Díky své nízké spotřebě a malým rozměrům je vhodný pro použití na mobilních zařízeních napájených z baterie. Senzor také umožňuje měření teploty a relativní vlhkosti. (https://cdn-shop.adafruit.com/)

Měření atmosférického tlaku funguje na principu piezorezistivního jevu, kde je měrný odpor výrazně závislý na mechanickém namáhání křemíkové destičky. (http://www.automa.cz)

Obrázek 9- Senzor BME280. *Zdroj: https://cdn-shop.adafruit.com/*

Rozsah a přesnost měření:

- teplota (-40 - 85) °C ± 1 °C,
- relativní vlhkost (0 - 100) % RH ± 3 % RH,
- tlak (300 - 1100) hPa ± 1 hPa.

Senzor BME280 je připojen pomocí I^2C sběrnice, která umožňuje komunikaci po dvou vodičích mezi procesorem (Master) a připojenou periferií (Slaves). Periferie je vybírána pomocí adresy, která je přenášena po stejných vodičích jako data. Sběrnice využívá datovou linku SDA a linku hodinového signálu SCL. (DH servis)

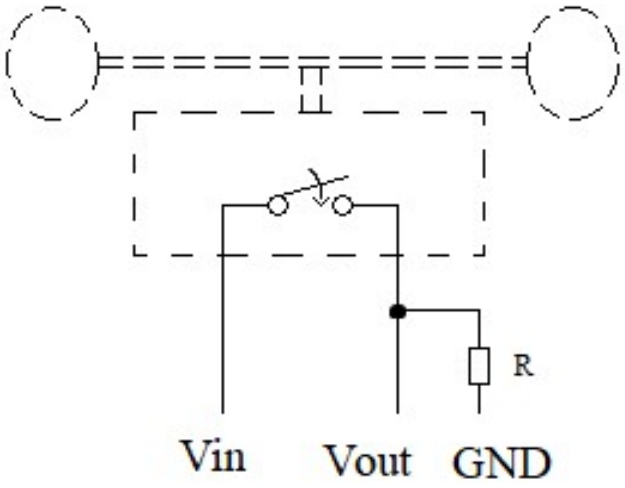
Senzor rychlosti větru

Pro měření rychlosti větru se využívá anemometr (Obrázek 10). Toto zařízení obsahuje vrtuli s lopatkami, na které vítr působí silou, a tím se vrtule otáčí. Uvnitř anemometru jsou dva spínače, které se vlivem otáčení neustále spínají a rozpínají. Pomocí počtu sepnutí spínačů je možné vypočítat průměrnou rychlost větru.

Obrázek 10- Anemometr. *Zdroj: https://www.hadex.cz/*

Senzor je k mikrokontroléru připojen pomocí tří vodičů (Obrázek 11). Pin Vin slouží k přivedení napětí do anemometru. Na pinu Vout jsou zaznamenávány změny napětí. Pin GND s pull-down rezistorem R odstraňuje šum, který vzniká na pinu Vout při rozepnutém spínači.

Obrázek 11- Schéma zapojení anemometru. *Zdroj: vlastní tvorba*



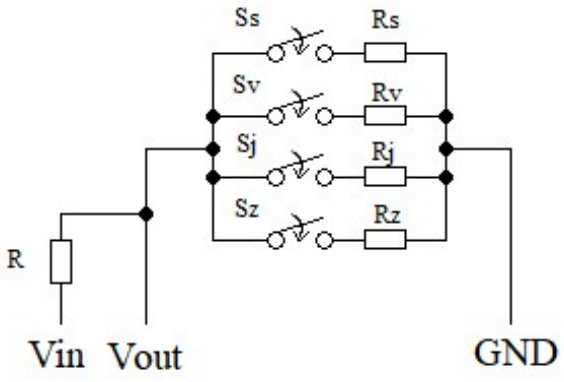
Senzor směru větru

Směr větru lze určovat pomocí dvou stupnic. Pomocí stupňů natočení od severu, nebo pomocí světových stran (sever, jih, východ, západ) a jejich kombinací. Senzor v sobě má několik spínačů, které se spínají a rozpínají v závislosti na natočení korouhvičky senzoru (Obrázek 12). Tyto spínače přepínají v obvodu paralelně zapojené rezistory, které mění výsledné napětí. Z úbytku napětí je následně možné dopočítat použité rezistory, a tím zjistit směr natočení.

Obrázek 12- Korouhvička. *Zdroj: https://www.hadex.cz/*

Rozsah senzoru závisí na počtu spínačů, v tomto případě má senzor čtyři spínače (Obrázek 13), kdy mohou být sepnuty vždy maximálně dva (sousední). Z toho vyplývá osm možných kombinací zapojení rezistorů (sever, severovýchod, východ, jihovýchod, jih, jihozápad, západ, severozápad).

Obrázek 13- Schéma zapojení rezistorů v korouhvičce. *Zdroj: vlastní tvorba*



Senzor je k mikrokontroléru připojen pomocí tří vodičů. Pin Vin s rezistorem R přivádí do senzoru napětí. Rezistor R spolu s rezistory Rs, Rv, Rj a Rz vytváří dělič napětí, který umožňuje měřit napětí na analogovém pinu Vout. Pin GND uzavírá obvod, a vytváří tak okruh, díky kterému může proud senzorem protékat.

Pro výpočet úbytku napětí můžeme využít vzorec pro dělič napětí, který vychází z Ohmova a Kirchhoffových zákonů.

$$U = U_R + U_K$$

$$U_K = U - U_R$$

$$U_K = U - I \cdot R$$

$$U_K = U - \frac{U}{R + R_K} \cdot R$$

$$U_K = \frac{U(R + R_K)}{R + R_K} - \frac{U \cdot R}{R + R_K}$$

$$U_K = \frac{U \cdot R + U \cdot R_K - U \cdot R}{R + R_K}$$

$$U_K = \frac{U \cdot R_K}{R + R_K}$$

Kde U je velikost vstupního napětí. R je odpor rezistoru R. R_K je odpor rezistoru v korouhvičce. I je celkový proud procházející obvodem. U_R je velikost napětí na rezistoru R. U_K je velikost napětí na rezistoru v korouhvičce.

Pokud nastane případ, ve kterém jsou sepnuty dva spínače, výsledný odpor R_K z rezistorů R_1 a R_2 vypočítáme ze vzorce pro paralelní zapojení rezistorů.

$$R_K = \frac{R_1 \cdot R_2}{R_1 + R_2}$$

Senzor DHT22

DHT22 (Obrázek 14) je kapacitní vlhkoměr a teploměr s velmi vysokou přesností. Všechny vyrobené kusy jsou testovány a kalibrovány ve speciální kalibrační komoře. Pro měření senzor využívá 8- bitový jednočipový počítač.

Princip měření relativní vlhkosti spočívá ve změně kapacity kondenzátoru. Ta je ovlivněna polymerním dielektrikem, které dokáže absorbovat vodní páry z ovzduší.

Měření teploty spočívá ve změně vodivosti polovodiče. Vodivost se mění s přibývající nebo ubývající teplotou. (https://cdn-shop.adafruit.com/)

Obrázek 14- Senzor DHT22. *Zdroj: https://core-electronics.com.au/*

Rozsah a přesnost měření:

- teplota (-40 - 80) °C ± 0,5 °C,
- relativní vlhkost (0 - 100) % RH ± 2 % RH.

Senzor je připojen k vývojové desce pomocí tří pinů. Na pin 1 je přivedeno vstupní napětí. Pin 2 je spojen s libovolným digitálním pinem, k tomuto datovému pinu je také připojen pull-up rezistor, který je na druhém konci připojen na vstupní napětí. Pull-up rezistor slouží k určování logické hodnoty na pinu 2 v době, kdy senzor žádná data neodesílá. Pin 3 je nepřipojený. Pin 4 je připojen ke GND, a uzavírá tak okruh.

3.3.2 Bezdrátová periferie

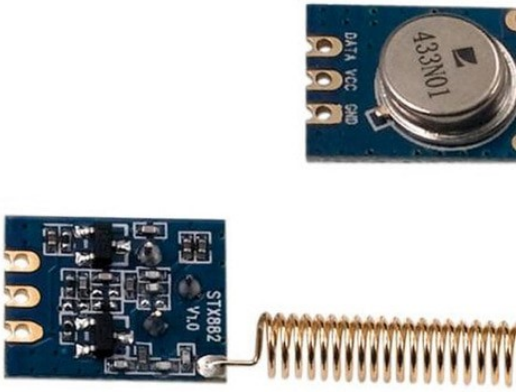
Umístění senzorů a čidel meteorologické stanice je stejně důležité jako jejich přesnost. Pokud dojde při měření ke zkreslení hodnot, pak získaná data nebudou odpovídat skutečnosti a jakákoliv předpověď spojená s nimi se stane chybnou. Komunikace venkovního a vnitřního čidla pomocí bezdrátové technologie výrazně přispívá k výběru lepšího umístění než propojení kabelem. Venkovní čidlo tak může být umístěno například na střeše, kde měření rychlosti a směru větru nebudou ovlivněny okolními stěnami.

Vysílač

Vysílač STX882 (Obrázek 15) je i přes svoji malou velikost nejvýkonnější vysílací zařízení v porovnání s ostatními vysílači se vstupním napětím do 3,6 V, jeho výkon je až 50 mW. I se svým výkonem poskytuje vysílač stabilní přenos dat. Modul jde využívat i při přímém zapojení k mikrokontroléru.

Důležitým příslušenstvím k vysílači je anténa. STX882 je konstruován pro využití antény se vstupní impedancí 50 Ω. (Smart-Prototyping)

Obrázek 15- Vysílač STX882 s anténou. *Zdroj: https://www.aliexpress.com/*



Vlastnosti vysílače:

- radiofrekvenční výkon: 20 dBm,
- spotřeba proudu při vysílání: 20 mA,
- spotřeba proudu při spánku: ≤ 0,01 μA,
- frekvence: 433 MHz,
- provozní teplota: (-20 - 70) °C.

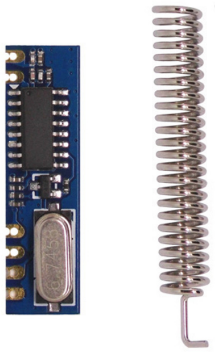
Modul lze propojit přímo s mikrokontrolérem Arduino a to pomocí tří vodičů. Pin Data slouží pro přenos vysílaných dat z mikrokontroléru. Pokud je na pinu detekováno nízké napětí (logická nula), vysílač se uspí, a výrazně tak sníží svoji spotřebu. Pin VCC je využíván pro napájení vysílače, napětí 5 V zvyšuje výkon vysílače, a umožnuje mu tak rozšířit jeho dosah. Pin GND slouží pro uzavření obvodu.

Přijímač

Přijímač SRX887 (Obrázek 16) je miniaturní, výkonově silný příjímací modul, který poskytuje vysokou stabilitu s velmi dobrým poměrem cena/efektivita. Modul je možné přímo připojit k mikrokontroléru.

Pro lepší efektivitu přijímání signálu muže být modul rozšířen o anténu se vstupní impedancí 50 Ω. (Plexishop.it)

Obrázek 16- Přijímač SRX887 s anténou. Zdroj: *https://www.aliexpress.com/*



Vlastnosti přijímače:

- citlivost: -107 dBm,
- spotřeba proudu při přijímání: 4 mA,
- spotřeba proudu při spánku: < 1 μA,
- frekvence: 433 MHz,
- provozní teplota: (-30 - 85) °C.

Modul lze propojit přímo s mikrokontrolérem Arduino a to pomocí čtyř vodičů. Pin VCC je využíván pro napájení přijímače. Pin Data slouží pro přenos přijímaných dat do mikrokontroléru. Pin CS lze využívat pro uspání a buzení přijímače. Pokud je pin CS připojen ke GND (na pinu CS je logická nula), tak přijímač nepřejde k režimu spánku a bude neustále v režimu provozu. Pin GND slouží pro uzavření obvodu.

3.3.3 Displej

Naměřené hodnoty spolu s jejich popiskem a jednotkami budou zobrazeny na displeji, který bude připojen na vnitřní čidlo. Proto by displej měl mít odpovídající rozlišení a barevnou hloubku, aby byl text dobře čitelný.

Displej ILI9340 (Obrázek 17) je TFT displej s tekutými krystaly, rozlišením 240 x 320 pixelů a velikosti 2,4 palce. S barevnou hloubkou 262 tisíc barev a LED posvícením je dobře čitelný i za tmy. Podsvícení lze vypnout a tím snížit spotřebu. (https://cdn-shop.adafruit.com/)

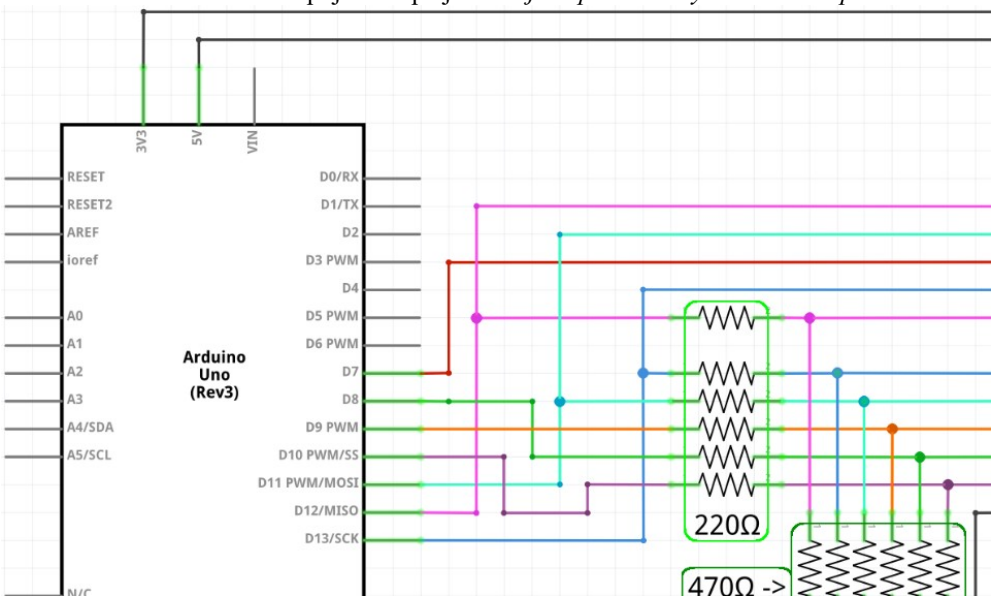
Obrázek 17- Displej ILI9341. *Zdroj: https://www.aliexpress.com/*

Pro zapojení displeje (Obrázek 18) je nutné propojit piny přes odporový dělič, který vytvoří napětí 3,3 V. V tomto případě se dělič skládá z rezistorů 220 Ω a 470 Ω. Piny MOSI, MISO a SCK pro displej a piny T_DIN, T_DO, T_CLK pro dotykovou vrstvu, slouží ke komunikaci pomocí rozhraní SPI.

Rozhraní SPI umožnuje komunikaci dvou či více propojených zařízení, kde jedno zařízení vystupuje jako řadič sběrnice (Master). Komunikace po sběrnici je zcela synchronní díky generátoru hodinového signálu SCK. Dalšími vodiči jsou MISO (Master IN, Slave OUT) a MOSI (Master OUT, Slave IN), které umožňují přenášení dat. Poslední vodič je CS, pomocí kterého je zařízení slave vybíráno. (Root, 1998)

Mikrokontroléry Arduino mají piny určené pro rozhraní SPI většinou rozdílné, například Arduino Mega využívá pro SPI rozhraní piny 50, 51, 52. (Arduino návody)

Obrázek 18- Zapojení displeje. *Zdroj: https://navody.arduino-shop.cz*



4 Vlastní práce

Tato část bakalářské práce obsahuje popis sestavení meteorologické stanice z vybraných komponentů a vývojových desek Arduino. Pro usnadnění zapojení komponentů k mikrokontrolérům Arduino je zde uveden návrh plošných spojů. Dále jsou zde popsány programy, které jsou nahrány ve venkovním a vnitřním čidle. V poslední části této kapitoly je provedena kalibrace senzorů.

4.1 Meteorologická stanice

Meteorologická stanice sestavená pro tuto bakalářskou práci využívá teoretické poznatky, které byly zmíněné v kapitole *3. Teoretická východiska.* Meteostanice se skládá ze dvou samostatných čidel, venkovního a vnitřního, které spolu bezdrátově komunikují.

Veličiny měřené venkovním čidlem:

- teplota,
- relativní vlhkost,
- atmosférický tlak,
- rychlost větru,
- směr větru.

Veličiny měřené vnitřním čidlem:

- teplota,
- relativní vlhkost.

Z teploty, relativní vlhkosti a rychlosti větru měřených na venkovním čidle jsou vypočítány hodnoty zdánlivé teploty a rosného bodu. Měřené hodnoty venkovní teploty jsou zaznamenávány až 24 hodin zpětně a je díky nim možné vykreslit graf vývoje teploty v čase.

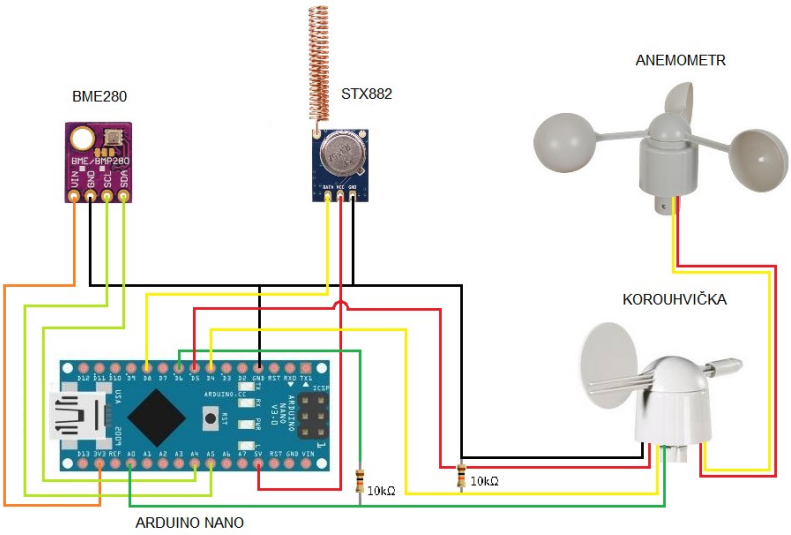
4.2 Návrh meteorologické stanice

Hardware meteorologické stanice je popsán v kapitolách *3.2 Mikrokontroléry Arduino* a *3.3 Komponenty*. V této části je vyobrazeno schéma zapojení komponentů k modelům Arduino.

4.2.1 Návrh zapojení venkovního čidla

Venkovní čidlo se skládá z vysílače STX882, senzoru BME280, anemometru a korouhvičky pro určení směru větru. Všechny tyto komponenty jsou připojeny k mikrokontroléru Arduino Nano. Schéma zapojení je zobrazeno na obrázku 19.

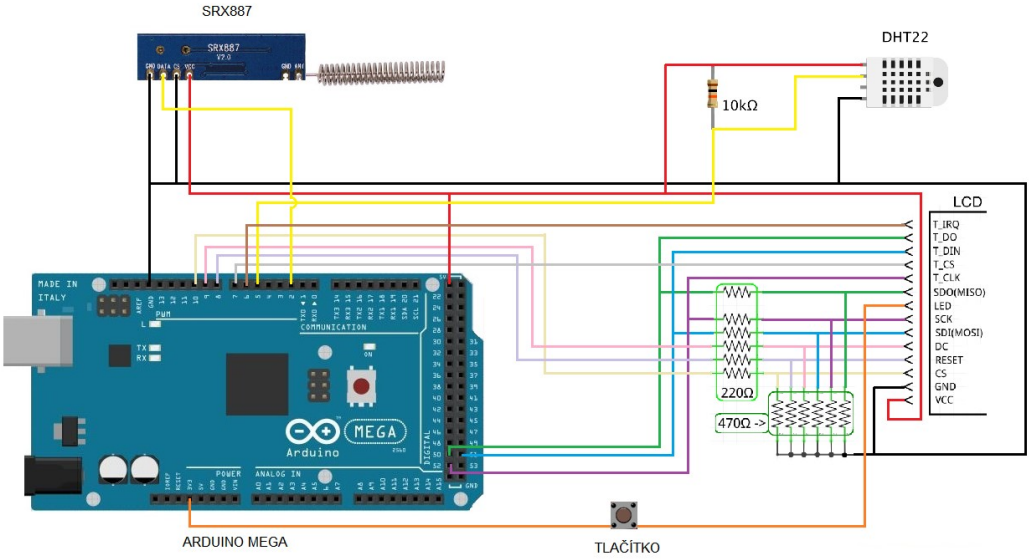
Obrázek 19- Zapojení venkovního čidla. *Zdroj: vlastní tvorba*



4.2.2 Návrh zapojení vnitřního čidla

Vnitřní čidlo je sestaveno z přijímače SRX887, senzoru DHT22 a LCD displeje. Rezistory o odporu 470Ω a 220Ω slouží jako dělič napětí pro displej. TLAČÍTKO přerušuje vodič napájení podsvícení displeje a slouží k úspoře energie. Schéma zapojení je vyobrazeno na obrázku 20.

Obrázek 20- Zapojení vnitřního čidla. *Zdroj: vlastní tvorba*



4.3 Návrh plošných spojů

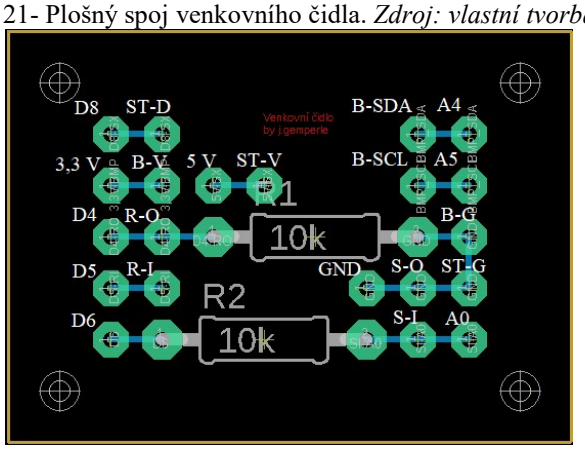
Plošný spoj je nevodivá destička, obsahující kovové spoje, které umožňují na daná místa připájet různé elektronické součástky, a usnadnit tak jejich připojení do elektrického obvodu. Pro návrh plošných spojů byl použit program Autodesk EAGLE.

Zelené kroužky slouží pro připájení elektronických součástek, jako jsou rezistory nebo konektory (v tomto případě kolíkové lišty). Modré čáry pak znázorňují kovové spoje mezi připájenými součástkami.

4.3.1 Plošný spoj pro venkovní čidlo

Plošný spoj (Obrázek 21) vyrobený pro venkovní čidlo je sestaven z propojených kolíkových lišt a rezistorů. Pin GND je společný pro všechny komponenty, které jsou napájeny od mikrokontroléru. Hodnoty rezistorů jsou R1 a R2 = 10kΩ

Obrázek 21- Plošný spoj venkovního čidla. *Zdroj: vlastní tvorba*



Způsob zapojení: (Arduino (PS) → (PS) Senzor/Modul)

Napětí 3,3 V (3,3 V) → (B-V) BMP280-VIN

Napětí 5 V (5 V) → (ST-V) STX882-VCC

GND → (S-O) Korouhvička output, (ST-G) STX882-GND, (B-G) BMP280-GND

Digitální pin 4 (D4) → (R-O) Anemometr output

Digitální pin 5 (D5) → (R-I) Anemometr input

Digitální pin 6 (D6), Analogový pin A0 (A0) → (S-I) Korouhvička input

Digitální pin 8 (D8) → (ST-D) STX882-Data

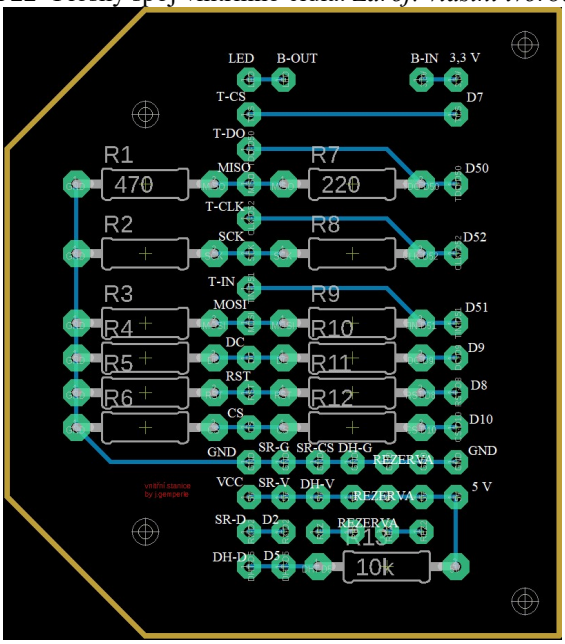
Analogový pin A4 (A4) → (B-SDA) BMP280-SDA

Analogový pin A5 (A5) → (B-SCL) BMP280-SCL

4.3.2 Plošný spoj pro vnitřní čidlo

Plošný spoj (Obrázek 22) vyrobený pro vnitřní čidlo je sestaven z propojených kolíkových lišt a rezistorů. Rezistory R1 – R12 na plošném spoji vytvářejí dělič napětí, díky kterému je vytvořeno napětí 3,3 V. Rezistor R13 slouží jako pull-up rezistor pro senzor DHT22. Hodnoty rezistorů jsou: R1-R6 = 470 Ω, R7-R12 = 220 Ω a R13 = 10kΩ

Obrázek 22- Plošný spoj vnitřního čidla. *Zdroj: vlastní tvorba*



Způsob zapojení: (Arduino (PS) → (PS) Senzor/Modul)

Napětí 3,3 V (3,3 V) → (B-IN) Tlačítko (B-OUT) → (LED) Displej led

Napětí 5 V (5 V) → (SR-V) SRX887-VCC, (DH-V) DHT22 pin1, (VCC) Displej vcc

GND → (SR-G) SRX887-GND, (SR-CS) SRX887-CS, (DH-G) DHT22 pin4

Digitální pin 2 (D2) → (SR-D) SRX887-DATA

Digitální pin 5 (D5) → (DH-D) DHR22 pin2

Digitální pin 7 (D7) → (T-CS) Displej t-cs

Digitální pin 8 (D8) → (RST) Displej rst

Digitální pin 9 (D9) → (DC) Displej dc

Digitální pin 10 (D10) → (CS) Displej cs

Digitální pin 50 (D50) → (T-DO) Displej t-do, (MISO) Displej miso

Digitální pin 51 (D51) → (T-IN) Displej t-in, (MOSI) Displej mosi

Digitální pin 52 (D52) → (T-CLK) Displej t-clk, (SCK) Displej sck

4.4 Program

Pro správnou funkčnost meteorologické stanice je nutné do mikrokontrolérů Arduino nahrát zkompilovaný program. Program se píše a kompiluje ve vývojovém prostředí Arduino IDE, které je volně ke stažení na webových stránkách.

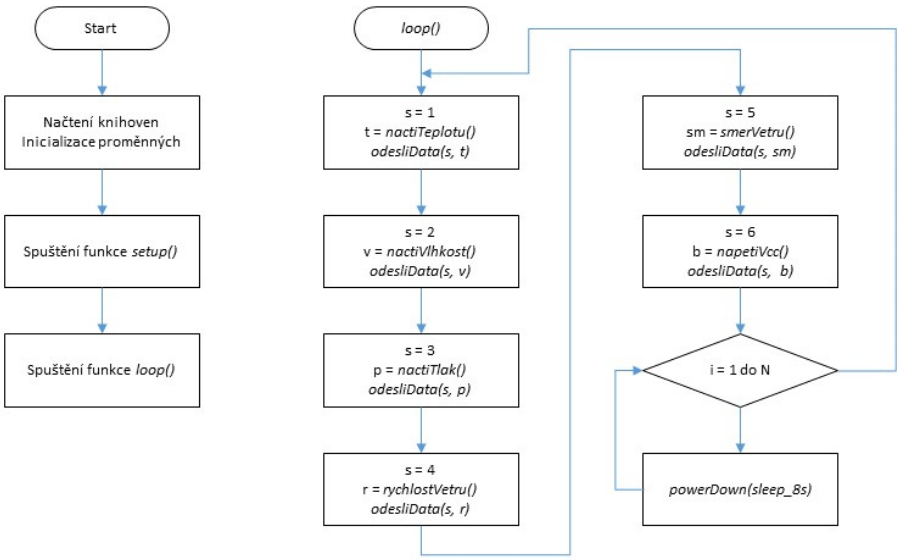
Arduino IDE umožňuje využívat knihovny pro různé komponenty, a tím výrazně zjednodušit práci s nimi. Knihovny obsahují již předdefinované metody právě pro práci s různými senzory a moduly.

Celé programy jsou uvedeny v kapitole *7 Přílohy.*

4.4.1 Program venkovního čidla

Program venkovního čidla slouží pro zjišťování měřených hodnot a jejich následné odesílání. Po proběhnutí jednoho cyklu, program uspí Arduino Nano, a tím výrazně sníží jeho spotřebu. Zjednodušený vývojový diagram programu je znázorněn na obrázku 23.

Obrázek 23- Vývojový diagram venkovního čidla. *Zdroj: vlastní tvorba*



Program začíná načtením knihoven pro senzor BME280, vysílač STX882 a uspávací knihovny. Dále jsou v programu inicializovány proměnné pro vstupní/výstupní piny, díky kterým je možné získávat nebo odesílat hodnoty, a konstanty pro adresu vysílače a počet opakování uspání.

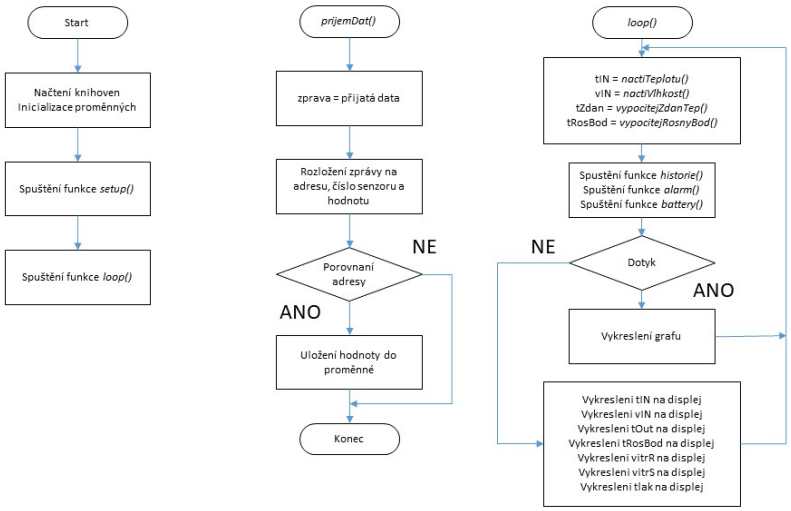
Následuje spuštění funkce *setup()*, která nastaví bezdrátovou komunikaci a naváže spojení se senzorem BME280. Po ukončení funkce *setup()* je spuštěna funkce *loop()*, ve které jsou postupně načítány a odesílány hodnoty z příslušných senzorů. Data jsou odesílána vždy ve formátu „*aaaashhhhh*“ (a = adresa, s = číslo senzoru, h = naměřená hodnota). Po odeslání všech dat následuje uspání mikrokontroléru. Díky knihovně *LowPower.h* je možné mikrokontrolér uspat pouze na 8 sekund. Proto je v programu vytvořen cyklus, který uspání opakuje až do požadovaného počtu.

Po ukončení uspávacího cyklu se program vrátí zpátky na začátek funkce *loop()*. Takto program pokračuje, dokud není mikrokontrolér odpojen od zdroje energie.

4.4.2 Program vnitřního čidla

Program vnitřního čidla slouží k přijímání měřených dat na venkovním čidle, měření vlastních dat na vnitřním čidle a zobrazování hodnot na displeji. Zjednodušený vývojový diagram programu je znázorněn na obrázku 24.

Obrázek 24- Vývojový diagram vnitřního čidla. *Zdroj: vlastní tvorba*



Program pro vnitřní čidlo začíná definováním potřebných knihoven a pinů pro displej, senzor DHT22 a přijímač SRX887. Dále jsou v kódu inicializovány proměnné pro hodnoty senzorů. Namíchané barvy pro grafické zpracování jsou uloženy v konstantách. Následuje inicializace proměnných pro naměřené hodnoty a vytvoření bitových polí obrázků značících stavy přijmu dat a baterie.

Následně je spuštěna funkce *setup()*, ve které je nastavena bezdrátová komunikace přijímače a zahájena komunikace s displejem a senzorem DHT22. Následuje vytvoření pozadí displeje a povolení přerušení pro funkci *prijemDat()*.

Funkce *prijemDat()* je spuštěna vždy, když je na zvoleném pinu detekována změna napětí. V takovém případě procesor přeruší stávající instrukce a začne vykonávat funkci *prijemDat()*, ve které jsou přijatá data uložena do proměnné *zprava*. Znakový řetězec je následně rozložen na části *adresa*, *senzor* a *hodnota*. Proměnná *adresa* slouží pro odfiltrování nechtěných zpráv, které nevysílá venkovní čidlo. Pomocí proměnné *senzor* je proměnná *hodnota* přiřazena do patřičné proměnné v programu, poté se funkce ukončí a procesor se vrátí k předešlým instrukcím.

Funkce *loop()* načte hodnoty ze senzoru DHT22, provede výpočty zdánlivé teploty a rosného bodu a zobrazí data na displeji. Funkce *historie()* ukládá hodnoty vybraných veličin, ze kterých jsou vykreslovány grafy. Záznam hodnot pokrývá 24 hodin. Funkce *alarm()* kontroluje aktuálnost přijatých dat. Pokud jsou data z venkovního čidla starší než pět minut, hodnoty zobrazené na displeji jsou překryty červenými obdélníky a v záhlaví displeje je zobrazena ikona alarmu komunikace. Funkce *battery()* ukazuje aktuální stav

baterie venkovního čidla pomocí hodnoty operačního napětí v mikrokontroléru. Všechny použité ikony jsou vyobrazeny na obrázku 25.

Obrázek 25- Ikony stavů baterie a příjmu dat. *Zdroj: vlastní tvorba*

Podmínka *dotyk* určuje, zda na displeji je vykreslen graf, nebo všechny hodnoty měřené meteorologickou stanicí.

4.5 Kalibrace

Kalibrace je soubor činností, kterými se za daných podmínek určí vztah mezi měřenými hodnotami a hodnotami odpovídající skutečnosti. Kalibrace tedy slouží k určení odchylky mezi skutečnou hodnotou a hodnotou měřenou senzorem.

4.5.1 Senzor rychlosti větru

Pro kalibraci senzoru rychlosti větru byl senzor umístěn na střechu auta. Z důvodu eliminování vlivu proudícího větru přes kapotu byl senzor od střechy vzdálen 70 cm. Povětrnostní podmínky v době kalibrace byly příznivé a rychlost větru, když auto stálo, byla 0 km/h.

Měření probíhalo v jedoucím autě, kde byla rychlost měřena systémem GPS a o její konstantní hodnotu se staral tempomat. Pro danou rychlost automobilu byl vždy zaznamenán počet impulsů v určeném časovém úseku. Měření bylo provedeno pro různé rychlosti. Naměřené hodnoty jsou zaznamenány v tabulce 1.

Tabulka 1- Kalibrace senzoru rychlosti větru. *Zdroj: vlastní tvorba*

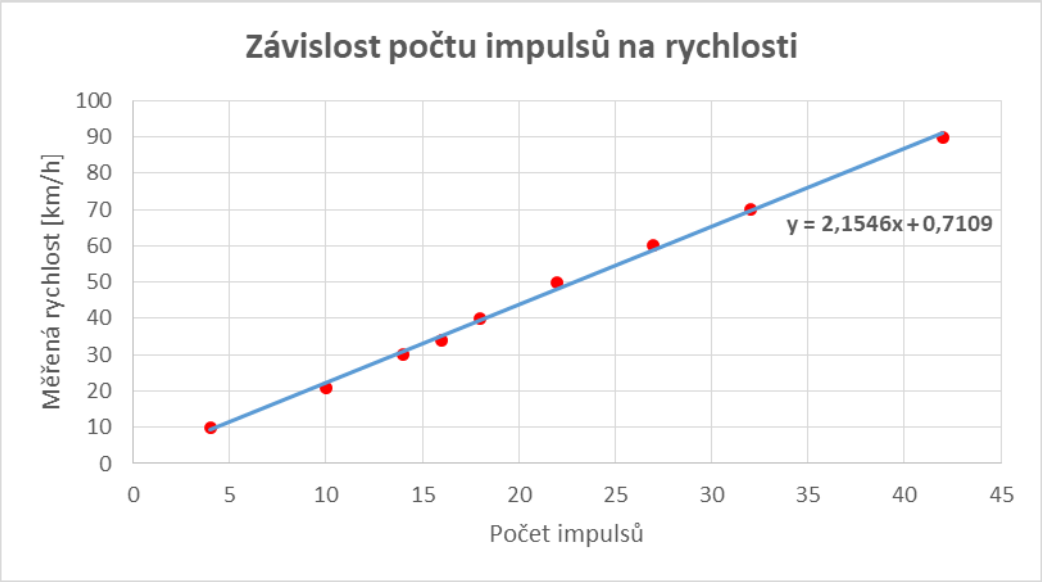
Měřená rychlost [km/h]	Počet impulsů	Vypočítaná rychlost [km/h]	Absolutní chyba [km/h]	Relativní chyba [%]
10	4	9,33	0,67	6,71
21	10	22,26	1,26	5,99
30	14	30,88	0,88	2,92
34	16	35,18	1,18	3,48
40	18	39,49	0,51	1,27
50	22	48,11	1,89	3,78
60	27	58,89	1,11	1,86
70	32	69,66	0,34	0,49
90	42	91,20	1,20	1,34

Závislost počtu impulsů na rychlosti je znázorněna v grafu na obrázku 26. Body grafu byla proložena přímka s rovnicí:

$$y = 2{,}1546 \cdot x + 0{,}7109$$

Absolutní chyba je učena rozdílem měřené rychlosti automobilu a vypočítané rychlosti pomocí rovnice přímky. Relativní chyba udává procentuální poměr absolutní chyby proti měřené rychlosti automobilu.

Obrázek 26- Závislost počtu impulsů na rychlosti. *Zdroj: vlastní tvorba*



4.5.2 Senzor směru větru

Senzor směru větru je kalibrován určením výsledného napětí, které je měřeno mikrokontrolérem. Pro výpočet napětí a odporu jsou použity vzorce zmíněné v kapitole *3.3.1 Senzory*. Výsledné hodnoty rezistorů a napětí jsou zaznamenány v tabulce 2.

Tabulka 2- Kalibrace senzoru směru větru. *Zdroj: vlastní tvorba*

Úhel [°]	Světová strana	Odpor v korouhvičce [Ω]	Odpor R [Ω]	Výstupní napětí [V]
0	Sever	33000	10000	3,84
45	Severovýchod	8200	10000	2,25
90	Východ	1000	10000	0,45
135	Jihovýchod	2200	10000	0,90
180	Jih	3900	10000	1,40
225	Jihozápad	16000	10000	3,08
270	Západ	120000	10000	4,62
315	Severozápad	64900	10000	4,33

Pro správnou funkci senzoru musí být korouhvička natočena ve správném směru. Pokud to instalace nedovoluje, k výslednému úhlu se připočítá rozdíl v natočení od severu.

5 Výsledky a diskuse

Sestavená meteostanice se skládá ze dvou čidel. Venkovní čidlo (Obrázek 27) je umístěno na střeše rodinného domu a měří venkovní teplotu, relativní vlhkost, atmosférický tlak a rychlost a směr větru. Hodnoty těchto měřených veličin následně odesílá pomocí vysílače.

Obrázek 27- Venkovní čidlo. *Zdroj: vlastní tvorba*

Vnitřní čidlo (Obrázek 28) měří vnitřní teplotu a relativní vlhkost. Kromě měření vnitřní čidlo také přijímá odeslané hodnoty z venkovního čidla, které spolu s vnitřními hodnotami vyobrazuje na displeji. Na displeji je také možné vykreslit graf vývoje venkovní teploty po dobu posledních 24 hodin (Obrázek 29).

Obrázek 28- Vnitřní čidlo. *Zdroj: vlastní tvorba*

Obrázek 29- Grafické zpracování hodnot. *Zdroj: vlastní tvorba*

Pro zjištění funkčnosti vytvořené meteorologické stanice byl proveden záznam měřených hodnot. Tyto hodnoty byly následně porovnány s hodnotami měřenými na certifikované meteorologické stanici ve městě Dobřichovice. Obě meteorologické stanice tedy byly umístěny ve stejné lokalitě. Měřené a ověřené hodnoty jsou vyobrazeny v tabulce 3.

Tabulka 3- Měřené a ověřené hodnoty. *Zdroj: vlastní tvorba a (Meteostanice, 2017e)*

	Měřené hodnoty			**Ověřené hodnoty**		
	Venkovní teplota [°C]	**Rosný bod [°C]**	**Atmosférický tlak [hPa]**	**Venkovní teplota [°C]**	**Rosný bod [°C]**	**Atmosférický tlak [hPa]**
1.	11,3	4,1	1019,1	9,7	3,4	1021,5
2.	11,4	4,0	1018,9	10,1	3,6	1021,1
3.	10,0	3,0	1018,6	9,0	2,8	1020,6
4.	8,7	5,8	1021,5	7,3	5,1	1022,9
5.	8,6	6,5	1022,3	7,6	5,6	1024,0
6.	9,1	6,5	1023,5	8,1	5,0	1025,3
7.	8,2	4,7	1025,1	6,8	3,9	1026,4
8.	7,2	2,4	1026,3	5,9	2,0	1027,1
9.	6,2	1,4	1027,5	5,0	1,1	1028,4
10.	5,9	0,9	1028,0	4,0	0,7	1029,2
11.	5,4	0,0	1029,1	3,6	-0,1	1030,2
12.	4,6	-1,2	1030,1	3,0	-1,7	1031,3
13.	4,2	-2,6	1030,0	2,9	-2,0	1032,0
14.	2,8	-3,6	1032,0	1,6	-3,7	1033,0
15.	1,9	-5,0	1033,2	1,0	-4,6	1033,9
16.	0,8	-5,3	1034,3	0,1	-5,3	1034,8
17.	0,4	-6,2	1035,0	-0,6	-5,6	1035,8

Kde ve sloupci *Měřené hodnoty* jsou zaznamenány hodnoty měřené na vytvořené meteorologické stanici. Ve sloupci *Ověřené hodnoty* jsou vypsány hodnoty získané z meteorologické stanice v Dobřichovicích.

5.1 Zhodnocení

Pro porovnání měřených a ověřených hodnot byla vypočítána absolutní chyba mezi hodnotami získanými ve stejný čas. Výsledky výpočtu jsou zaznamenány v tabulce 4. Absolutní chyba je vypočítána jako absolutní hodnota rozdílu měřené a ověřené hodnoty.

Tabulka 4- Absolutní chyba. *Zdroj: vlastní tvorba*

	Absolutní chyba		
	Venkovní teplota [°C]	**Rosný bod [°C]**	**Atmosférický tlak [hPa]**
1.	1,6	0,7	2,4
2.	1,3	0,4	2,2
3.	1,0	0,2	2,0
4.	1,4	0,7	1,4
5.	1,0	0,9	1,7
6.	1,0	1,5	1,8
7.	1,4	0,8	1,3
8.	1,3	0,4	0,8
9.	1,2	0,3	0,9
10.	1,9	0,2	1,2
11.	1,8	0,1	1,1
12.	1,6	0,5	1,2
13.	1,3	0,6	2,0
14.	1,2	0,1	1,0
15.	0,9	0,4	0,7
16.	0,7	0,0	0,5
17.	1,0	0,6	0,8

Rozdíl mezi měřenými a ověřenými hodnotami může být dán kvalitou senzorů ve vyrobené meteorologické stanici. Celková cena vyrobené meteorologické stanice se pohybuje okolo 3 000 Kč. Pokud by byly komponenty koupeny v zahraničních online obchodech, cena by byla ještě výrazně nižší.

Z toho lze usoudit, že profesionální meteorologické stanice s pořizovací cenou i desítky tisíc korun českých budou poskytovat přesnější měření. Cílem této bakalářské práce však bylo vytvořit meteorologickou stanici pro domácí využití a pro ni je přesnost měřených hodnot podle mého názoru dostatečná.

6 Závěr

Hlavními cíli této bakalářské práce bylo vytvořit meteorologickou stanici na platformě Arduino, využít bezdrátovou technologii pro komunikaci mezi venkovním a vnitřním čidlem a porovnat hodnoty měřené vytvořenou meteorologickou stanicí s hodnotami získanými z profesionální meteostanice. Všechny tyto cíle byly úspěšně naplněny.

První část práce se zabývá definicemi a základními pojmy v oblasti meteorologických stanic, jejich historií, představením mikrokontrolérů Arduino a komponentů, které lze k mikrokontrolérům připojit. Meteorologické stanice je možné rozdělit do několika skupin, a to podle zaměření, umístění a získávaných dat. Je možné získávat údaje o teplotě, vlhkosti vzduchu, tlaku, slunečním svitu, rychlosti a směru větru. Platforma Arduino se skládá z velkého počtu modelů, z nichž jsou v této práci použity dva. Prvním je model Arduino Mega 2560 a druhým Arduino Nano. Hlavní rozdíl mezi těmito modely spočívá ve velikosti flash paměti a počtu vstupních/výstupních pinů.

Vlastní práce se nachází ve druhé části. Meteorologická stanice byla vytvořena ze dvou čidel. Pro venkovní čidlo byl použit model Arduino Nano. Čidlo měří teplotu, relativní vlhkost, atmosférický tlak, rychlost větru a směr větru. Naměřené hodnoty posílá ke druhému čidlu, které využívá model Arduino Mega 2560. Čidla spolu bezdrátově komunikují. Spolu s měřením vnitřní teploty a relativní vlhkosti vnitřní čidlo zobrazuje naměřené a přijaté hodnoty na displeji. Hodnoty venkovní teploty je čidlo schopné vykreslit do grafu, a zobrazit tak vývoj teploty za posledních 24 hodin. Pro zjednodušení zapojení komponentů k mikrokontrolérům byly vytvořeny plošné spoje. Programy pro venkovní i vnitřní čidlo byly napsány na základě nakreslených vývojových diagramů.

Poslední část obsahuje zhodnocení naměřených výsledků. Hodnoty měřené venkovním čidlem byly porovnány s hodnotami měřenými certifikovanou meteorologickou stanicí v Dobřichovicích. Pro porovnání měřených a získaných hodnot byla vypočítána absolutní chyba, která zobrazuje rozdíl mezi hodnotami.

7 Seznam použitých zdrojů

- *DH servis* Vývoj a výroba elektroniky na zakázku [online]. [cit. 29.01.2019]. Dostupné z: http://www.dhservis.cz/iic.htm
- [online]. Copyright © [cit. 29.01.2019]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/ILI9340.pdf
- [online]. Copyright ©I [cit. 25.01.2019]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/Digital+humidity+and+temperature+sensor+AM2302.pdf
- [online]. Copyright ©X [cit. 25.01.2019]. Dostupné z: http://www.automa.cz/Aton/FileRepository/pdf_articles/42719.pdf
- [online]. [cit. 29.01.2019]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf
- [online]. [cit. 29.01.2019]. Dostupné z: http://www.pocasi.astronomie.cz/aerologicky_vystup.htm
- [online]. [cit. 29.01.2019]. Dostupné z: https://www.tomperys.cz/stav-baterie-s-arduinem/?lang=en
- [online]. [cit. 29.01.2019]. Dostupné z: https://vetrani.tzb-info.cz/teorie-a-vypocty-vetrani-klimatizace/3137-vlhkost-vzduchu-a-jeji-mereni
- Arduino Mega 2560 Rev3. Arduino [online]. © a [cit. 29.01.2019]. Dostupné z: https://store.arduino.cc/arduino-mega-2560-rev3
- Arduino Nano. Arduino [online]. © b [cit. 29.01.2019]. Dostupné z: https://store.arduino.cc/arduino-nano
- BELL, Charles A. Beginning sensor networks with Arduino and Raspberry Pi. New York, New York: Apress, 2013. ISBN 1430258241.
- Displej dotykový 240x320 px | Arduino návody. Webový magazín o ARDUINU | Arduino návody [online]. [cit. 29.01.2019]. Dostupné z: https://navody.arduino-shop.cz/navody-k-produktum/displej-dotykovy-240x320.html
- Externí sériové sběrnice SPI a I²C - Root.cz. Root.cz - informace nejen ze světa Linuxu [online]. Copyright © 1998 [cit. 08.02.2019]. Dostupné z: https://www.root.cz/clanky/externi-seriove-sbernice-spi-a-i2c/
- Meteorologické stanice - dělení a význam | In-počasí. Počasí - předpověď počasí, aktuální informace | In-počasí [online]. Copyright © 2017a [cit. 09.12.2018].

Dostupné z: https://www.in-pocasi.cz/clanky/teorie/meteorologicke-stanice-rozdeleni/

- Meteostanice - Dobřichovice, aktuální teplota, vítr, tlak, srážky | In-počasí. Počasí - předpověď počasí, aktuální informace | In-počasí [online]. Copyright © 2017e [cit. 23.02.2019]. Dostupné z: https://www.in-pocasi.cz/meteostanice/stanice.php?stanice=dobrichovice
- Plexishop.it - Plexishop!, L'arte del plexiglass! [online]. Copyright © [cit. 27.01.2019]. Dostupné z: https://www.plexishop.it/pdf/SRX887.pdf
- Praha Klementinum - meteorologická stanice | In-počasí. Počasí - předpověď počasí, aktuální informace | In-počasí [online]. Copyright © 2017b [cit. 09.12.2018]. Dostupné z: https://www.in-pocasi.cz/archiv/klementinum.php
- Rosný bod a jeho význam v meteorologii | In-počasí. Počasí - předpověď počasí, aktuální informace | In-počasí [online]. Copyright © 2017d [cit. 26.01.2019]. Dostupné z: https://www.in-pocasi.cz/clanky/teorie/rosny-bod-20.2.2015/
- Sluneční svit a jeho měření | In-počasí. Počasí - předpověď počasí, aktuální informace | In-počasí [online]. Copyright © 2017c [cit. 10.12.2018]. Dostupné z: https://www.in-pocasi.cz/clanky/teorie/slunecni-svit/
- Smart-Prototyping [online]. Copyright ©Gv [cit. 27.01.2019]. Dostupné z: https://www.smart-prototyping.com/image/data/9_Modules/101415%20STX882%20ASK%20Transmitter%20Module/STX882%20Datasheet.pdf
- Stručně o meteorologických stanicích. Meteorologická stanice Maruška - Hostýnské vrchy [online]. Copyright © 2006 [cit. 09.12.2018]. Dostupné z: http://maruska.ordoz.com/meteorologie/meteorologicke_stanice
- Thermal Comfort observations. Australia's official weather forecasts & weather radar - Bureau of Meteorology [online]. [cit. 29.01.2019]. Dostupné z: http://www.bom.gov.au/info/thermal_stress/index.shtml
- Víte, co jsou to meteorologické radiosondy? | Teorie | Aktuality o počasí | Počasícz.cz. Počasí, dlouhodobá předpověď počasí pro ČR | Počasícz.cz [online]. [cit. 29.01.2019]. Dostupné z: https://www.pocasicz.cz/aktuality-o-pocasi/teorie-2451/vite-co-jsou-to-meteorologicke-radiosondy-2804

8 Přílohy

8.1 Program venkovního čidla

```
#include "VirtualWire.h"
#include "LowPower.h"
#include "Adafruit_BME280.h"

#define BME280_ADRESA (0x76)
Adafruit_BME280 bme;

#define rychlostPIN 4
#define rychlostOutputPIN 5
#define smerOutputPIN 6
#define txPIN 8

#define smerPIN A0

#define dobaMereniVetru 1

char adresa[] = "00x1";
int interval = 3;

void setup() {
  pinMode(rychlostPIN, INPUT);
  pinMode(rychlostOutputPIN, OUTPUT);
  pinMode(smerPIN, INPUT);
  pinMode(smerOutputPIN, OUTPUT);

  if (!bme.begin(BME280_ADRESA)) {
    Serial.println("BME280 senzor nenalezen, zkontrolujte zapojeni!");
    while (1);
  }

  vw_set_ptt_inverted(true);
  vw_setup(1000);
  vw_set_tx_pin(txPIN);

  Serial.println("-- Weather Station Scenario --");
  Serial.println("forced mode, 1x temperature / 1x humidity / 1x pressure oversampling,");
  Serial.println("filter off");
  bme.setSampling(Adafruit_BME280::MODE_FORCED,
                  Adafruit_BME280::SAMPLING_X1, // temperature
                  Adafruit_BME280::SAMPLING_X1, // pressure
                  Adafruit_BME280::SAMPLING_X1, // humidity
                  Adafruit_BME280::FILTER_OFF   );
  delay(1000);
}
```

```
void loop() {
  digitalWrite(rychlostOutputPIN, HIGH);
  digitalWrite(smerOutputPIN, HIGH);

  byte senzor;
  bme.takeForcedMeasurement();

  float t = nactiTeplotu();
  senzor = 1;
  odesliData(senzor, t);

  float v = nactiVlhkost();
  senzor = 2;
  odesliData(senzor, v);

  float p = nactiTlak();
  senzor = 3;
  odesliData(senzor, p);

  float r = rychlostVetru();
  senzor = 4;
  odesliData(senzor, r);

  float s = smerVetru();
  senzor = 5;
  odesliData(senzor, s);

  float b = napetiVcc();
  senzor = 6;
  odesliData(senzor, (float) b);

  digitalWrite(rychlostOutputPIN, LOW);
  digitalWrite(smerOutputPIN, LOW);
  delay(100);

  for (int i = 0; i < interval; i++) {
    LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
  }
}

float nactiTeplotu() {
  float tep = bme.readTemperature();
  return tep;
}

float nactiVlhkost() {
  float vl = bme.readHumidity();
  return vl;
}
```

```
float nactiTlak() {
  float tl = bme.readPressure();
  return (tl / 100 + 17.6);
}

float rychlostVetru() {
  int pocetImpulsu = 0;
  int stavImpulsu;
  int stavImpulsuM;
  unsigned long pocatecniCas;
  float rych;
  pocatecniCas = millis();
  while (millis() <= pocatecniCas + dobaMereniVetru * 1000)
  {
    stavImpulsu = digitalRead(rychlostPIN);
    delay(2);
    if (stavImpulsu != stavImpulsuM && stavImpulsu == LOW)
    {
      pocetImpulsu++;
    }
    stavImpulsuM = stavImpulsu;
  }
  if (pocetImpulsu > 0) {
    rych = (float)(2.1546 * pocetImpulsu / dobaMereniVetru + 0.7109) / 3.6;
  } else {
    rych = 0;
  }
  return rych;
}

float smerVetru() {
  float odpory[8] = {33000, 8200, 1000, 2200, 3900, 16000, 120000, 64900};
  float mezNapeti[8];
  float vcc = napetiVcc();
  float hodnota = analogRead(smerPIN);
  float napeti = (float) hodnota * vcc / 1023;
  for (int i = 0; i < 8; i++) {
    mezNapeti[i] = (float) vcc * odpory[i] / (10000 + odpory[i]) + 0.1;
  }
  float sm;
  if (napeti <= mezNapeti[2])
    sm = 90;
  else if (napeti <= mezNapeti[3])
    sm = 135;
  else if (napeti <= mezNapeti[4])
    sm = 180;
  else if (napeti <= mezNapeti[1])
    sm = 45;
  else if (napeti <= mezNapeti[5])
```

```
    sm = 225;
  else if (napeti <= mezNapeti[0])
    sm = 0;
  else if (napeti <= mezNapeti[7])
    sm = 315;
  else
    sm = 270;

  return sm;
}

float napetiVcc() {
 #if defined(__AVR_ATmega32U4__) || defined(__AVR_ATmega1280__) ||
 defined(__AVR_ATmega2560__)
  ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
 #elif defined (__AVR_ATtiny24__) || defined(__AVR_ATtiny44__) ||
 defined(__AVR_ATtiny84__)
  ADMUX = _BV(MUX5) | _BV(MUX0);
 #elif defined (__AVR_ATtiny25__) || defined(__AVR_ATtiny45__) ||
 defined(__AVR_ATtiny85__)
  ADMUX = _BV(MUX3) | _BV(MUX2);
 #else
  ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
 #endif

  delay(2);
  ADCSRA |= _BV(ADSC);
  while (bit_is_set(ADCSRA, ADSC));

  int stavBaterie;
  uint8_t low = ADCL;
  uint8_t high = ADCH;

  long napetiMV = (high << 8) | low;

  napetiMV = 1125300L / napetiMV; // Výpočet Vcc (mV); 1125300 = 1.1*1023*1000
  float napetiV = (float) napetiMV / 1000;
  return napetiV;
}

void odesliData(byte senzor, float hodnota) {
  char msg[VW_MAX_MESSAGE_LEN];
  strcpy(msg, adresa);
  itoa(senzor, msg + strlen(msg), DEC);
  dtostrf(hodnota, 4, 2, msg + strlen(msg));
  vw_send((uint8_t *)msg, strlen(msg));
  vw_wait_tx();
  delay(10);
}
```

8.2 Program vnitřního čidla

```
#include "VirtualWire.h"
#include "SPI.h"
#include "Math.h"
#include "Adafruit_ILI9340.h"
#include "Adafruit_GFX.h"
#include "XPT2046_Touchscreen.h"
#include "DHT.h"

#define LCD_cs 10
#define LCD_dc 9
#define LCD_rst 8
#define DOTYK_cs 7
#define DOTYK_irq 6

#define dhtPIN 5
#define dhtTYPE DHT22
DHT dht(dhtPIN, dhtTYPE);

#define rxPIN 2
String adresa = "00x1";

uint16_t pozadiB, ramecekB, darkBlueB, lightBlueB, alarmB;

float tOut = NAN;
float vOut = NAN;
float tlak = NAN;
float vitrR = NAN;
float vitrS = NAN;
float vcc = NAN;

float tOutLast = NAN;
float vOutLast = NAN;
float tlakLast = NAN;
float vitrRLast = NAN;
float vitrSLast = NAN;

float tZdanLast, tRosBodLast;
int imgVcc;

unsigned long dataLast;
unsigned long cas;

boolean dataAlarm = false;
boolean modeGraph = false;
boolean modeT = false;
boolean modeV = false;
```

```
boolean modeTL = false;
boolean mode = true;

float tInLast, vInLast;

float hTOUT[232];
float hRV[232];
float hTlak[232];
int velHis = sizeof(hTOUT) / sizeof(hTOUT[0]) - 2;

Adafruit_ILI9340 displej = Adafruit_ILI9340(LCD_cs, LCD_dc, LCD_rst);
XPT2046_Touchscreen dotyk(DOTYK_cs);

extern uint8_t data[];
extern uint8_t baterie0[];
extern uint8_t baterie1[];
extern uint8_t baterie2[];
extern uint8_t baterie3[];

void setup() {
  vw_set_ptt_inverted(true);
  vw_setup(1000);
  vw_set_rx_pin(rxPIN);
  vw_rx_start();

  attachInterrupt(0, prijemDat, CHANGE);

  displej.begin();
  dotyk.begin();
  displej.setRotation(0);
  pozadiB = displej.Color565(0, 51, 102);
  ramecekB = displej.Color565(0, 61, 100);
  darkBlueB = displej.Color565(0, 77, 125);
  lightBlueB = displej.Color565(51, 153, 204);
  alarmB = displej.Color565(204, 51, 51);
  displej.fillScreen(pozadiB);

  displej.fillRect(4, 0, 160, 32, ramecekB);
  displej.setTextColor(ILI9340_WHITE);
  displej.setTextSize(2);
  displej.setCursor(12, 8);
  displej.println("METEOSTANICE");

  pozadiMeteo();

  dht.begin();

  delay(1000);
}
```

```
void loop() {
  float tIN = nactiTeplotu();
  float vIN = nactiVlhkost();
  float tZdan = vypocitejZdanTep();
  float tRosBod = vypocitejRosnyBod();

  historie();
  alarm();
  battery();

  if (!modeGraph) {
    if (tIN <= tInLast - 0.2 || tIN >= tInLast + 0.2 || !mode) {
      tInLast = tIN;
      senzorNaDisplej(152, 36, 84, 32, 156, 44, tIN, 216, 52, "s.C", darkBlueB);
    }

    if (vIN <= vInLast - 1 || vIN >= vInLast + 1 || !mode) {
      vInLast = vIN;
      senzorNaDisplej(152, 72, 84, 32, 156, 80, vIN, 216, 88, "%", darkBlueB);
    }

    if ((tOut != tOutLast && !isnan(tOut)) || !mode) {
      tOutLast = tOut;
      senzorNaDisplej(152, 108, 84, 32, 156, 116, tOut, 216, 124, "s.C", lightBlueB);
    }

    if ((tZdan != tZdanLast && !dataAlarm && !(isnan(tOut) || isnan(vOut) ||
isnan(vitrR))) || !mode) {
      tZdanLast = tZdan;
      senzorNaDisplej(152, 144, 84, 32, 156, 152, tZdan, 216, 160, "s.C", darkBlueB);
    }

    if ((tRosBod != tRosBodLast && !dataAlarm && !(isnan(tOut) || isnan(vOut))) ||
!mode) {
      tRosBodLast = tRosBod;
      senzorNaDisplej(152, 180, 84, 32, 156, 188, tRosBod, 216, 196, "s.C", darkBlueB);
    }

    if ((vitrR != vitrRLast && !isnan(vitrR)) || !mode) {
      vitrRLast = vitrR;
      senzorNaDisplej(152, 216, 84, 32, 156, 224, vitrR, 216, 232, "m/s", lightBlueB);
    }

    if ((vitrS != vitrSLast && !isnan(vitrS)) || !mode) {
      vitrSLast = vitrS;
      senzorNaDisplej(152, 252, 84, 32, 156, 260, (int) vitrS, 216, 268, "s.", darkBlueB);
    }
```

```
if ((tlak != tlakLast && !isnan(tlak)) || !mode) {
     tlakLast = tlak;
     senzorNaDisplej(116, 288, 120, 32, 120, 296, tlak, 216, 304, "hPa", lightBlueB);
   }
   mode = true;

  } else if (modeGraph && modeT) {

   senzorNaDisplej(152, 216, 84, 32, 156, 224, hTOUT[velHis + 1], 216, 232, "s.C",
lightBlueB);

   senzorNaDisplej(152, 252, 84, 32, 156, 260, hTOUT[velHis - 1], 216, 268, "s.C",
lightBlueB);

   senzorNaDisplej(152, 288, 84, 32, 156, 296, hTOUT[velHis], 216, 304, "s.C",
lightBlueB);

   int xPos = 4;
   float graphHight;
   for (int i = 0; i < velHis; i++) {
     graphHight = map(hTOUT[i], hTOUT[velHis] - 5, hTOUT[velHis + 1] + 5, 0, 174);
     displej.drawFastVLine(xPos, 211 - graphHight, graphHight, ILI9340_WHITE);
     xPos++;
   }

   modeT = false;

  } else if (modeGraph && modeV) {

   senzorNaDisplej(152, 216, 84, 32, 156, 224, hRV[velHis + 1], 216, 232, "m/s",
lightBlueB);

   senzorNaDisplej(152, 252, 84, 32, 156, 260, hRV[velHis - 1], 216, 268, "m/s",
lightBlueB);

   senzorNaDisplej(152, 288, 84, 32, 156, 296, hRV[velHis], 216, 304, "m/s", lightBlueB);

   int xPos = 4;
   float graphHight;
   for (int i = 0; i < velHis; i++) {
     graphHight = map(hRV[i], hRV[velHis] - 5, hRV[velHis + 1] + 5, 0, 174);
     displej.drawFastVLine(xPos, 211 - graphHight, graphHight, ILI9340_WHITE);
     xPos++;
   }

   modeV = false;

  } else if (modeGraph && modeTL) {
```

```
    senzorNaDisplej(116, 216, 120, 32, 120, 224, hTlak[velHis + 1], 216, 232, "hPa",
lightBlueB);

    senzorNaDisplej(116, 252, 120, 32, 120, 260, hTlak[velHis - 1], 216, 268, "hPa",
lightBlueB);

    senzorNaDisplej(116, 288, 120, 32, 120, 296, hTlak[velHis], 216, 304, "hPa",
lightBlueB);

    int xPos = 4;
    float graphHight;
    for (int i = 0; i < velHis; i++) {
      graphHight = map(hTlak[i], hTlak[velHis] - 5, hTlak[velHis + 1] + 5, 0, 174);
      displej.drawFastVLine(xPos, 211 - graphHight, graphHight, ILI9340_WHITE);
      xPos++;
    }

    modeTL = false;
  }

  if (dotyk.touched() && digitalRead(DOTYK_irq) == LOW) {
    stiskTlacitka();
  }
  delay(10);
}

float nactiTeplotu() {
  float tep = dht.readTemperature();
  return tep;
}

float nactiVlhkost() {
  float vl = dht.readHumidity();
  return vl;
}

float vypocitejRosnyBod() {
  float citatel = (243.5 * log(vOut * exp(17.67 * tOut / (243.5 + tOut)) / 100));
  float jmenovatel = (17.67 - log(vOut * exp(17.67 * tOut / (243.5 + tOut)) / 100));
  float rb = citatel / jmenovatel;
  return rb;
}

float vypocitejZdanTep() {
  float vt = vOut * 6.105 * exp(17.27 * tOut / (237.7 + tOut)) / 100;
  float zt = tOut + 0.33 * vt - 0.70 * vitrR - 4;
  return zt;
}
```

```
void historie() {
  if ((millis() > cas + 372000) && !dataAlarm) {
    if (cas != 0) {
      hTOUT[velHis] = tOut;
      hTOUT[velHis + 1] = tOut;
      hRV[velHis] = vitrR;
      hRV[velHis + 1] = vitrR;
      hTlak[velHis] = tlak;
      hTlak[velHis + 1] = tlak;

      for (int i = 0; i < velHis - 1; i++) {
        hTOUT[i] = hTOUT[i + 1];
        hTOUT[velHis] = min(hTOUT[velHis], hTOUT[i]);
        hTOUT[velHis + 1] = max(hTOUT[velHis + 1], hTOUT[i]);

        hRV[i] = hRV[i + 1];
        hRV[velHis] = min(hRV[velHis], hRV[i]);
        hRV[velHis + 1] = max(hRV[velHis + 1], hRV[i]);

        hTlak[i] = hTlak[i + 1];
        hTlak[velHis] = min(hTlak[velHis], hTlak[i]);
        hTlak[velHis + 1] = max(hTlak[velHis + 1], hTlak[i]);
      }
      hTOUT[velHis - 1] = tOut;
      hRV[velHis - 1] = vitrR;
      hTlak[velHis - 1] = tlak;
    } else {
      for (int i = 0; i < velHis + 2; i++) {
        hTOUT[i] = tOut;
        hRV[i] = vitrR;
        hTlak[i] = tlak;
      }
    }
    cas = millis();
  } else if (millis() < cas) {
    cas = millis();
  }
}

void alarm() {
  if ((millis() >= dataLast + 300000) || dataLast == 0) {
    if (!dataAlarm) {
      displej.drawBitmap(204, 0, data, 32, 32, alarmB, ILI9340_WHITE);
      displej.fillRect(204, 0, 32, 2, alarmB);
      alarmNaDisplej();
      dataAlarm = true;
    }
  } else if (dataAlarm) {
    displej.drawBitmap(204, 0, data, 32, 32, darkBlueB, ILI9340_WHITE);
```

```
    displej.fillRect(204, 0, 32, 2, darkBlueB);
    dataAlarm = false;
  } else if (millis() < dataLast) {
    dataLast = millis();
  }
}

void battery() {
  if (!dataAlarm && vcc < 3.9 && imgVcc != 0) {
    displej.drawBitmap(168, 0, baterie0, 32, 32, alarmB, ILI9340_WHITE);
    displej.fillRect(168, 0, 32, 2, alarmB);
    imgVcc = 0;
  } else if (!dataAlarm && vcc >= 3.9 && vcc < 4.3 && imgVcc != 1) {
    displej.drawBitmap(168, 0, baterie1, 32, 32, darkBlueB, ILI9340_WHITE);
    displej.fillRect(168, 0, 32, 2, darkBlueB);
    imgVcc = 1;
  } else if (!dataAlarm && vcc >= 4.3 && vcc < 4.6 && imgVcc != 2) {
    displej.drawBitmap(168, 0, baterie2, 32, 32, darkBlueB, ILI9340_WHITE);
    displej.fillRect(168, 0, 32, 2, darkBlueB);
    imgVcc = 2;
  } else if (!dataAlarm && vcc >= 4.6 && imgVcc != 3) {
    displej.drawBitmap(168, 0, baterie3, 32, 32, darkBlueB, ILI9340_WHITE);
    displej.fillRect(168, 0, 32, 2, darkBlueB);
    imgVcc = 3;
  }
}

void senzorNaDisplej(int rX, int rY, int rS, int rV, int cX, int cY, float h, int jX, int jY, String
j, uint16_t barva) {
  displej.setTextSize(2);
  displej.fillRect(rX, rY, rS, rV, barva);
  displej.setCursor(cX, cY);
  displej.println(h, 1);
  displej.setTextSize(1);
  displej.setCursor(jX, jY);
  displej.println(j);
}

void senzorNaDisplej(int rX, int rY, int rS, int rV, int cX, int cY, int h, int jX, int jY, String j,
uint16_t barva) {
  displej.setTextSize(2);
  displej.fillRect(rX, rY, rS, rV, barva);
  displej.setCursor(cX, cY);
  displej.println(h);
  displej.setTextSize(1);
  displej.setCursor(jX, jY);
  displej.println(j);
}
```

```
void alarmNaDisplej() {
  displej.fillRect(152, 108, 84, 32, alarmB);
  displej.fillRect(152, 144, 84, 32, alarmB);
  displej.fillRect(152, 180, 84, 32, alarmB);
  displej.fillRect(152, 216, 84, 32, alarmB);
  displej.fillRect(152, 252, 84, 32, alarmB);
  displej.fillRect(116, 288, 120, 32, alarmB);
  tOut = NAN;
  vOut = NAN;
  tlak = NAN;
  vitrR = NAN;
  vitrS = NAN;
  vcc = NAN;
  imgVcc = NAN;
  cas = 0;
}

void prekreslitPozadi() {
  displej.fillRect(0, 32, 240, 288, pozadiB);

}

void pozadiMeteo() {
  displej.setTextSize(1);
  displej.setCursor(8, 48);
  displej.println("Vnitrni teplota:");
  displej.setCursor(8, 84);
  displej.println("Vnitrni vlhkost:");
  displej.setCursor(8, 120);
  displej.println("Venkovni teplota:");
  displej.setCursor(8, 156);
  displej.println("Zdanliva teplota:");
  displej.setCursor(8, 192);
  displej.println("Rosny bod:");
  displej.setCursor(8, 228);
  displej.println("Rychlost vetru:");
  displej.setCursor(8, 264);
  displej.println("Smer vetru:");
  displej.setCursor(8, 300);
  displej.println("Tlak:");
}

void pozadiGraf(String s) {
  displej.setTextSize(1);
  displej.setCursor(8, 228);
  displej.println("MAXimalni " + s);
  displej.setCursor(8, 264);
  displej.println("AKTualni " + s);
  displej.setCursor(8, 300);
```

```
  displej.println("MINimalni " + s);
  displej.drawRect(4, 36, 232, 176, ILI9340_WHITE);
}

void stiskTlacitka() {
  TS_Point bod = dotyk.getPoint();
  if (bod.x > 2450 && bod.x < 2750 && bod.y > 340 && bod.y < 1501 && !dataAlarm
&& !modeGraph) {
    prekreslitPozadi();
    pozadiGraf("teplota:");
    modeGraph = true;
    modeT = true;
  } else if (bod.x > 1236 && bod.x < 1510 && bod.y > 340 && bod.y < 1501 &&
!dataAlarm && !modeGraph) {
    prekreslitPozadi();
    pozadiGraf("vitr:");
    modeGraph = true;
    modeV = true;
  } else if (bod.x > 100 && bod.x < 740 && bod.y > 340 && bod.y < 2084 &&
!dataAlarm && !modeGraph) {
    prekreslitPozadi();
    pozadiGraf("tlak:");
    modeGraph = true;
    modeTL = true;
  } else if (bod.x > 100 && bod.x < 1510 && bod.y > 340 && bod.y < 1501 &&
!dataAlarm && modeGraph) {
    prekreslitPozadi();
    pozadiMeteo();
    modeGraph = false;
    mode = false;

  }
}

void prijemDat() {
  uint8_t zprava[VW_MAX_MESSAGE_LEN];
  uint8_t delkaZpravy = VW_MAX_MESSAGE_LEN;
  if (vw_get_message(zprava, &delkaZpravy)) {
    String msg = zprava;
    msg.remove(delkaZpravy);

    if (msg.substring(0, adresa.length()).compareTo(adresa) == 0) {
      int senzor = msg.substring(adresa.length(), adresa.length() + 1).toInt();
      float hodnota = msg.substring(adresa.length() + 1).toFloat();
      switch (senzor) {
        case 1:
          tOut = hodnota;
          break;
        case 2:
```

```
        vOut = hodnota;
        break;
      case 3:
        tlak = hodnota;
        break;
      case 4:
        vitrR = hodnota;
        break;
      case 5:
        vitrS = hodnota;
        break;
      case 6:
        vcc = hodnota;
        break;
    }
    dataLast = millis();
  }
 }
}
```