Středoškolská technika 2016

Setkání a prezentace prací středoškolských studentů na ČVUT

Konstrukce meteostanice jako úvod ke studiu IoT

Vladimír Váňa, Martin Hrubý, Anja Gasparjan, Adam Švehla

Střední průmyslová škola elektrotechnická
Ječná 30, Praha 2

Úvod

Cílem práce je navrhnout a realizovat meteostanici sloužící pro výuku studentů oboru informační technologie a to jako zdroj dat při výuce databázových systémů, datových skladů, cloudů apod. Dalším důvodem je to, že se přitom zároveň naučí používat stejné mikroelektronické prvky a programování jejich firmware, které použije i v oblasti IoT a později ve firemní praxi např. při řešení projektů pro Průmysl 4 (mikrokontroléry, čidla, bezdrátovou komunikaci malým výkonem).

1. Kapitola

Soukromé meteostanice.

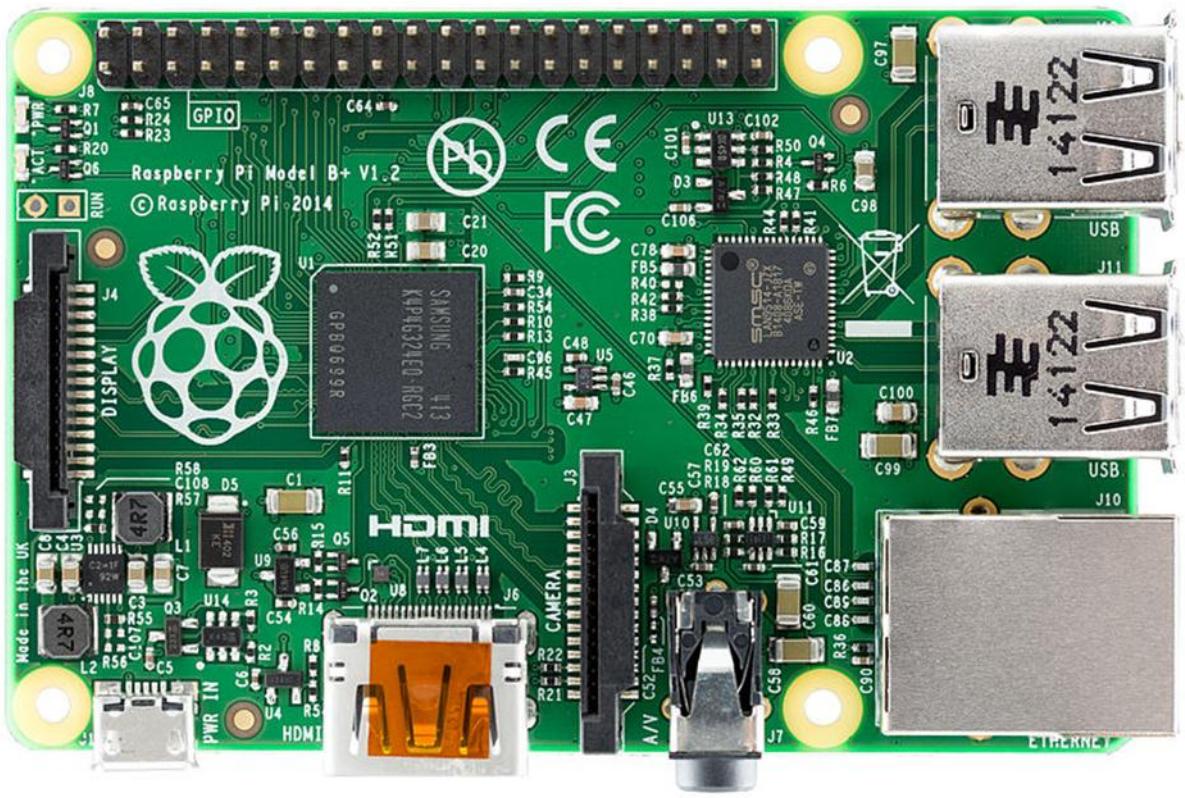
Měření veličin charakterizujících počasí jako je např. teplota a záznam naměřených hodnot probíhá již několik století. Provádělo se a provádí v institucích jako jsou univerzity, ústavy akademie věd, leteckých či lodních dopravců apod. Používaly se přitom měřící přístroje využívající mechanické, optické, tepelné či chemické vlastnosti látek. Příkladem může být lihový či rtuťový teploměr, vlhkoměr využívající závislost délky lidských vlasů na vlhkosti, membranový tlakoměr apod.

V minulém století se rozšířila i spolupráce amaterských pozorovatelů počasí s univerzitami či ústavy akademie, které je vybavily meteorologickými stanicemi. Ty byly umístěny na jejich soukromých pozemcích jako třeba na zahradě a amaterští pozorovatelé k nim několikrát denně docházeli a zapisovali naměřená dat, která pak předávali příslušné instituci. Ta takto získala síť pozorovacích stanic, což ji umožnilo např. získat mapu s meteorologickou situací na daném území a z vývoje změn počasí se pak např. snažit o předpověď počasí. V dobách před existencí meteorologických satelitů to byla vlastně i jediná možnost, jak získat přehled o meteorologické situaci na určitém území. Kromě amaterských spolupracovníkú ústavů měli některé jednoduché měřící přiístroje k dispozici i další soukromé osoby. Typicky byly ve vlastnictví soukromých osob lihové či rtuťové teploměry pro měření vnitřní či venkovní teploty. Méně častěji pak měřiče atmosférického tlaku či vlkosti nebo měření či indikaci směru a rychlosti větru.

Situace se však podstatně změnila s rozvojem mikroelektroniky, která umožňuje výrobu jednoduchých a laciných meteostanic vybavených jednočipovým počítačem a řadou různých čidel včetně čidel MEMS, bezdrátovým přenosem naměřených veličin mezi venkovními čidly a meteostanicí umístěných v bytě. Na nich se pak kromě naměřených veličin a času (např. z DCF77) často zobrazuje i časový průběh naměřených veličin, východ a západ slunce a měsíce a předpověď počasí. Cena takových meteostanic začíná na několika stokorunách a cena těch nejdražších bohatě vybavených nepřesahuje 5.000 Kč. Tyto meteostanice jsou určeny především soukromým osobám, ale používají je např. i školy a naměřené údaje publikují na školním webu. Naskýtá se proto otázka, proč jsme vyvíjeli meteostanici vlastní, když meteostanici s podobnými vlastnostmi lze zakoupit.
Důvod je jednoduchý. Umožní nám to naučit se pracovat s technologiemi v mohutně se rozvíjející oblasti IoT (Internet of Things). V oblasti IoT se často využívají bezdrátové sítě čidel. Jejich nódy obsahují nějaký mikrokontrolér, který má jako periferie řadu čidel. Bezdrátově jsou propojeny k centrálnímu počítači. Jím může být např Raspberry PI, Arduino, Nucleo, PicAXE apod. Tento počítač může být připojen k internetu. Obdobně jako v případě IoT lze pro uchování, zpracování a presentaci dat použít nějaký cloud.

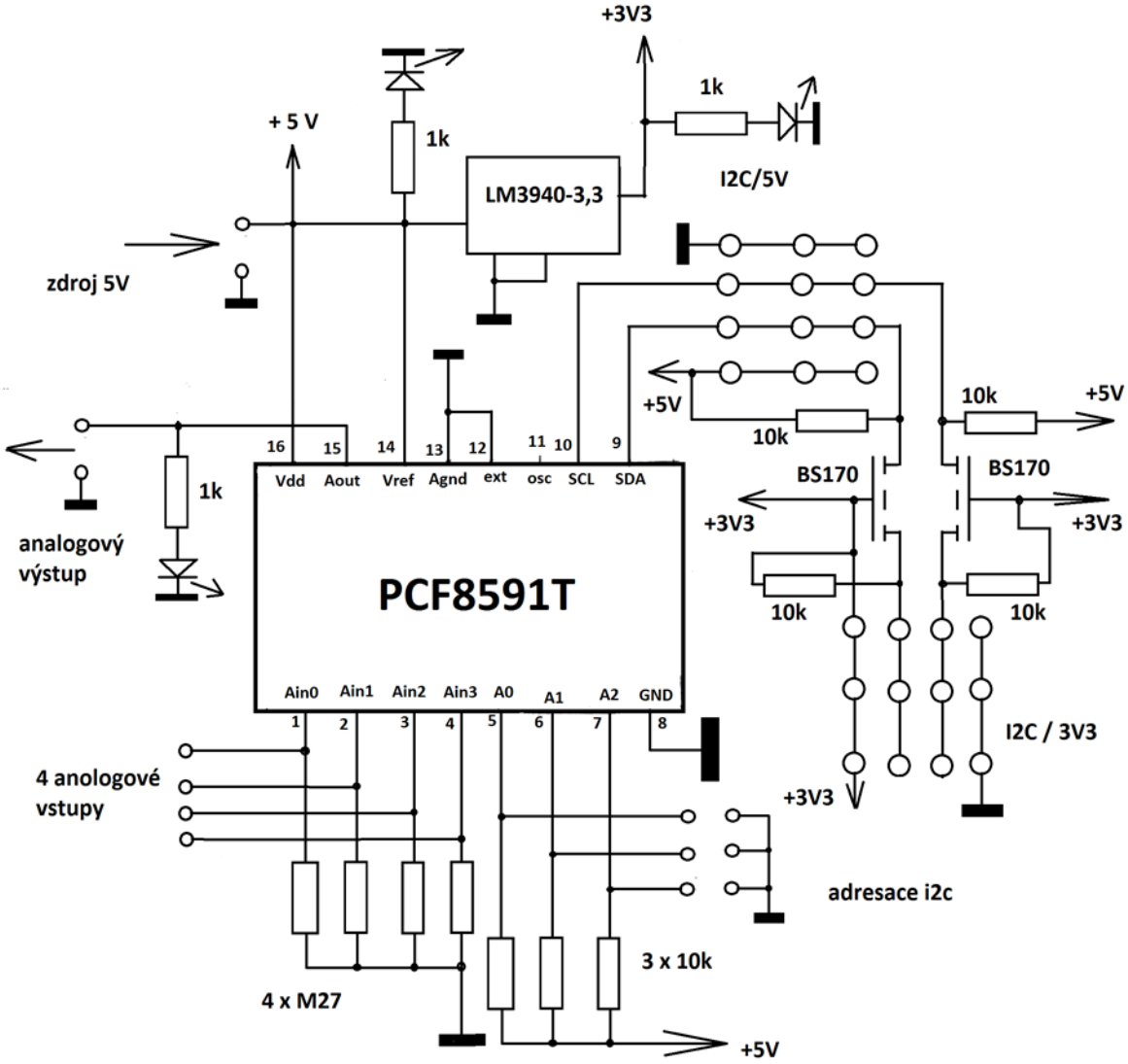
1.2. Raspberry PI

Pro meteostanici jsme využili tento levný počítač.

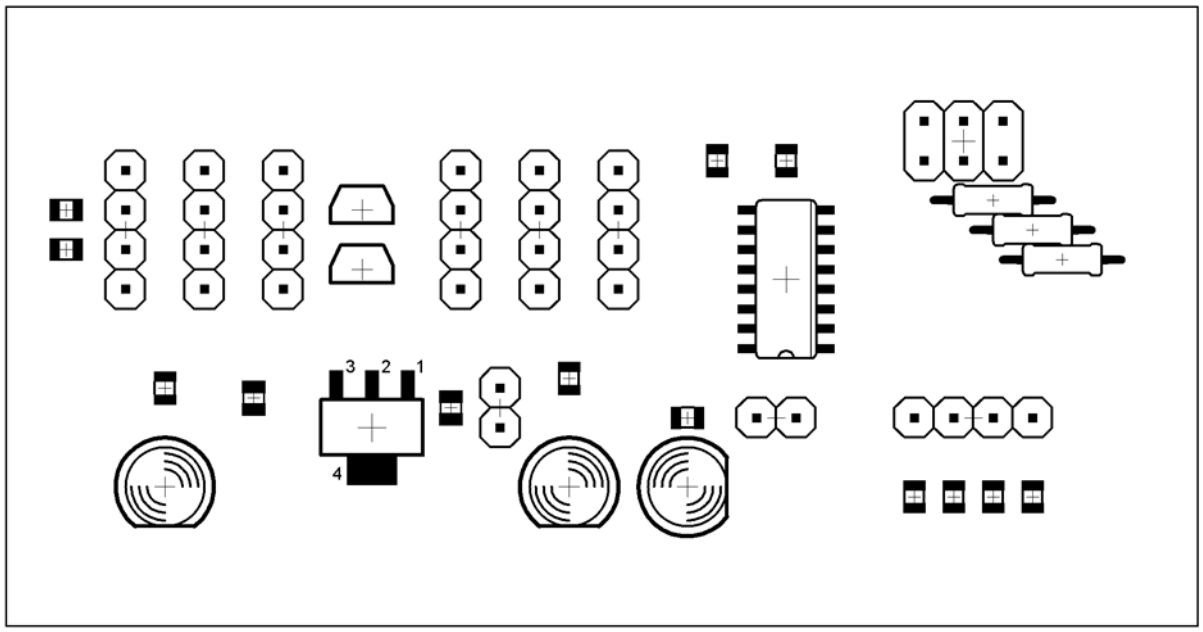


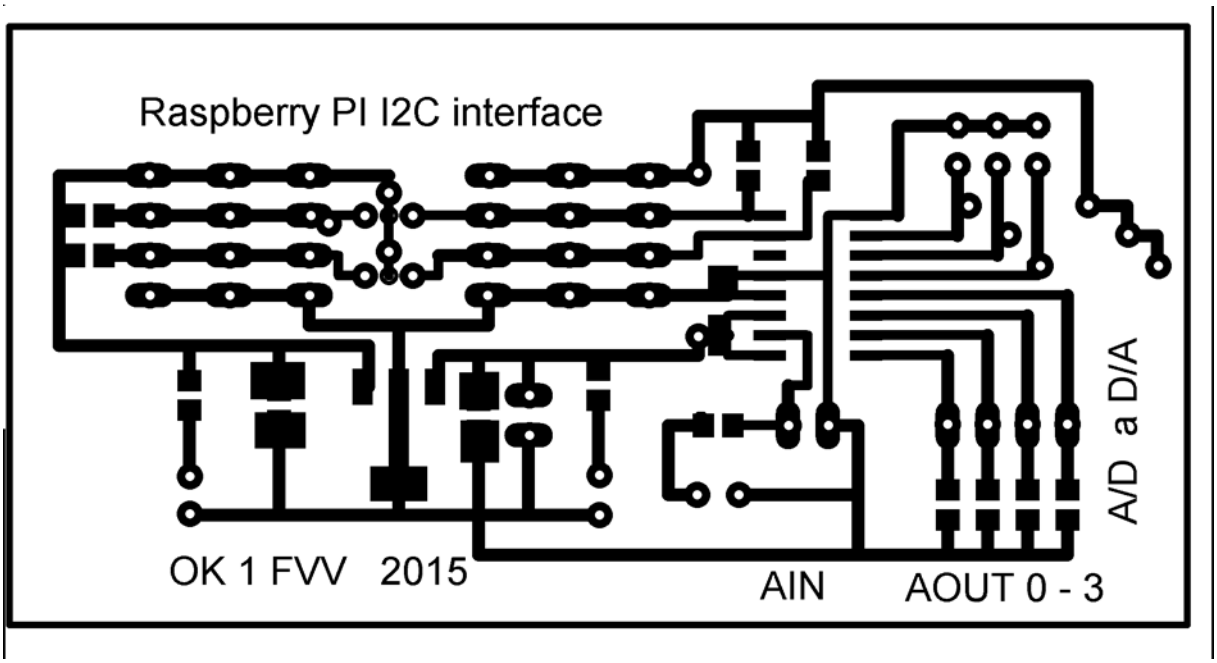
V současné době existují dvě významné skupiny mikrořadičů (mikrokontrolérů, jednočipových počítačů) lišících se tím, s jakými logickými úrovněmi pracují. Ty starší jsou navrženy pro TTL úrovně, tj. 5V logiku, ty novější pak pro úroveň 3V3 (3.3V). K nim existuje i řada periferních obvodů, např. různých čidel. Do skupiny pracující s 5V logikou patří např. Arduino, 3V3 má např. Raspberry PI či NUCLEO s obvody STM32. Třebaže

STM32 pracuje s logikou 3V3 jsou jeho I/O piny schopné pracovat s 5V signálem. Bohužel Raspberry PI lze 5V signálem poškodit. Navíc MCU Raspberry PI neobsahuje vestavěné A/D a D/A převodníky. Lze ale použít externí převodníky připojené k RPI pomocí I2C. Proto jsme navrhli a realizovali desku obsahující zdroj 3V3 realizovaný obvodem LM3940IMP-3.3 , dále dvojici obousměrných převodníků úrovní 5V – 3V3 realisovanou tranzistory řízenými polem BS170 a sloužící pro převod signálů SDA a SCL sběrnice I2C a dále čtveřici 8bitových AD převodníků spolu s jedním 8bitovým DA převodníkem integrované v obvodu PCF8591T komunikujícího s I2C. Schema této desky ukazuje následující obrázek:



Rozložení součástí a PCB (použit sw Eagle) pak obrázky následující:



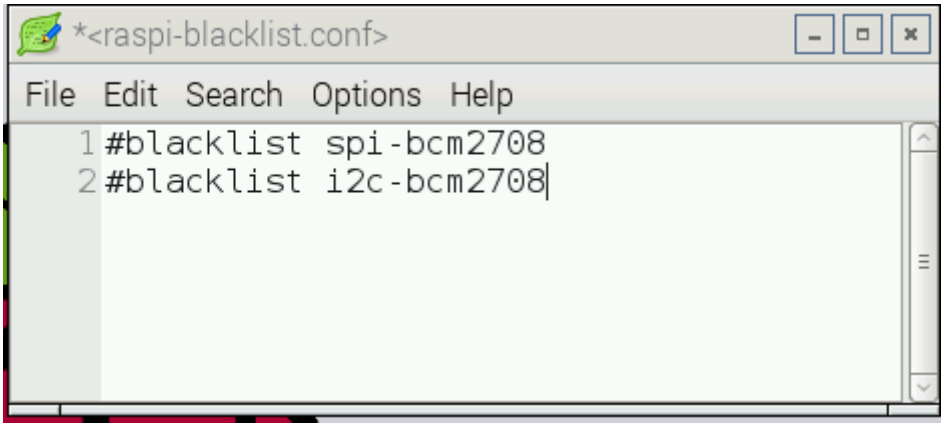


Ještě uvedeme fotografie hotové desky:

Tuto desku propojíme s RPI pomocí 3 vodičů – GND, dále SDA na GPIO2 a SDA na GPIO3.Pro zprovoznění I2C ještě provedeme (popř. zkontrolujeme zda je již provedeno) následující úpravy v OS: Nejdříve je potřeba povolit ovladače sběrnice I2C –

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

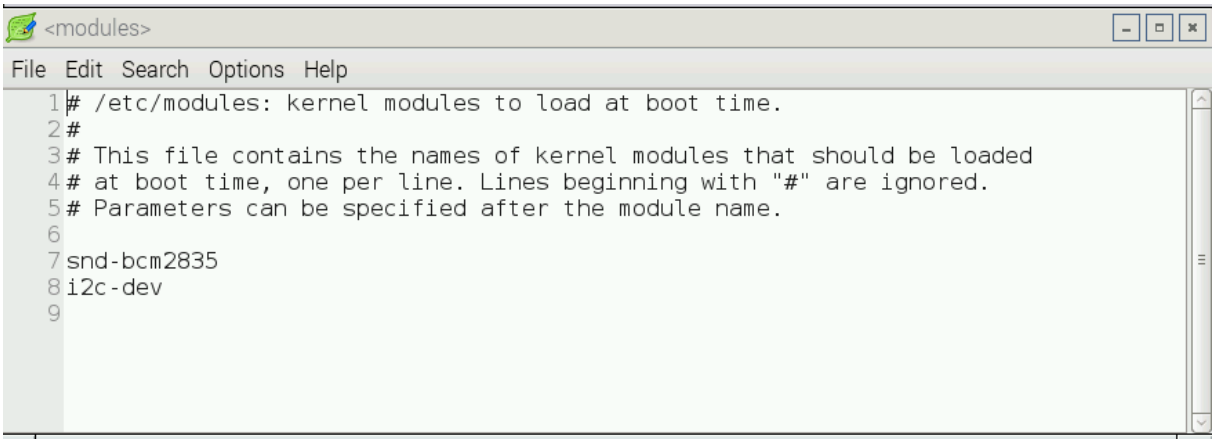
V editoru, který se tímto příkazem otevře, se musí zakomentovat dvě řádky (připsat před ně #), jak je vidět na následujícím obrázku :



a změny uložit (Ctrl+X, Y, Enter). Pak nastavit modul I2C, aby se spouštěl automaticky po startu Rasberry Pi - V editoru otevřít soubor **/etc/modules**

```
sudo nano /etc/modules
```

A do něj na konec dopsat řádku: **i2c-dev**



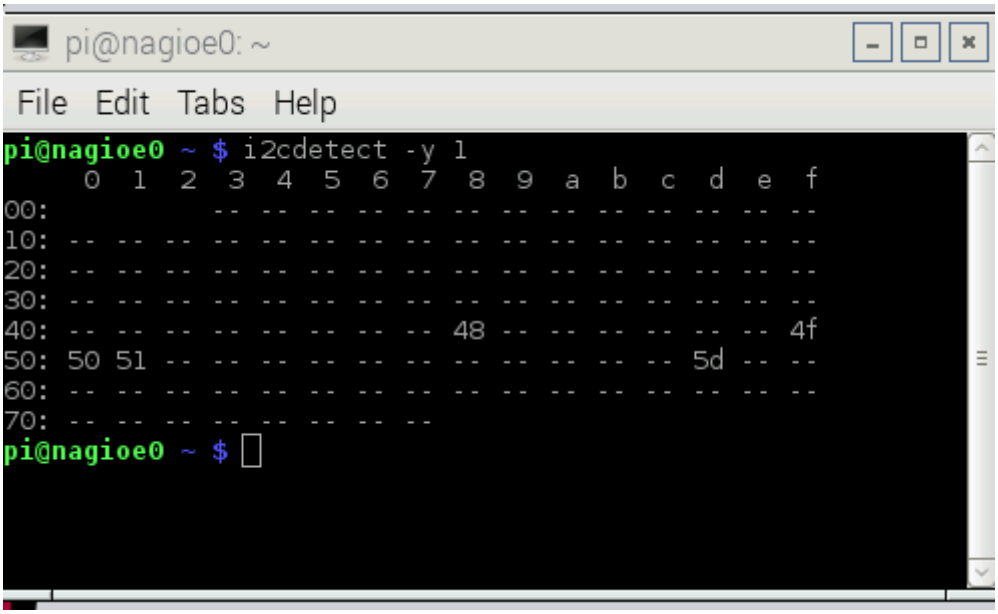
Nakonec restartovat:

```
sudo reboot
```

Nyní napíšeme příkaz

i2cdetect –y 1

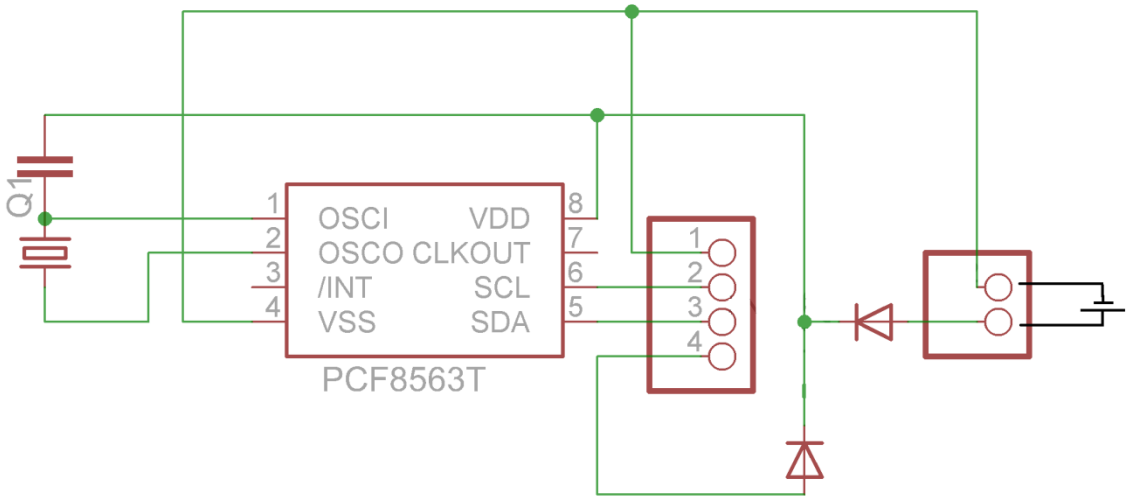
Měl by se objevit výpis adres připojených i2c zařízení, např.

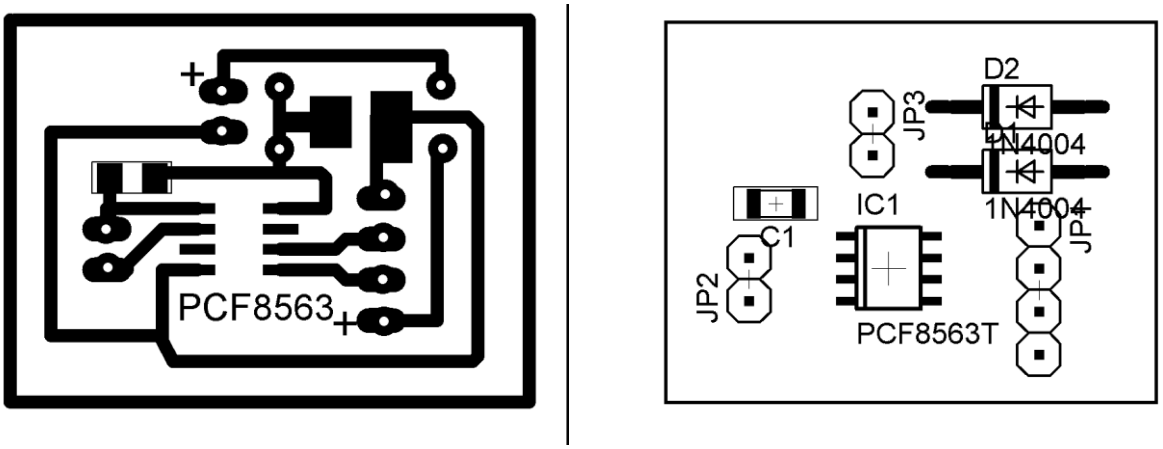


pozn.
V našem případě máme na adrese 0x48 obvod PCF8591T (4 x AD převodník a jeden DA), na adr. 0x4f čidlo teploty AD7416ARMZ, které jsme umístili na malé destičce spolu s čidlem atmosférického tlaku LPS331AP, který má adresu 0x5d. Adresu 0x50 má obvod PCF853, který máme zapojený jako čítač impulzů z anemometru T114 sloužícího jako čidlo rychlosti větru. Adresa 0x51 patří obvodu reálného času (hodiny, kalendář) PCF8563.

1. 3. Obvod kalendáře a reálného času

Pokud budeme naměřená data dále zpracovávat, zaznamenávat a zobrazovat jejich časové průběhy, potřebujeme znát i údaje o čase měření. RPI nemá zabudovaný obvod RTC . V případě připojení RPI k internetu lze využít časové servery, problém může nastat v případě jejich poruchy či pokud není RPI připojen k internetu . Proto jsme navrhli a realizovali jednoducou destičku s obvodem PCF8563 který komunikuje prostřednictvím i2c.

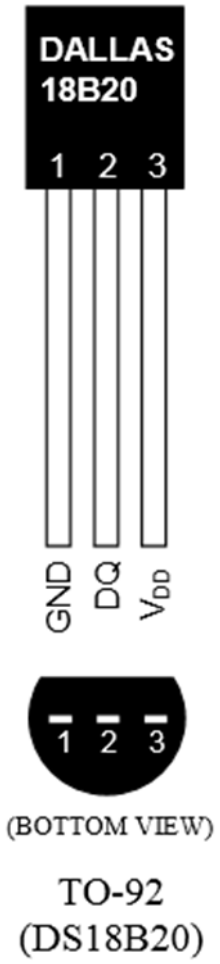




2. Kapitola - čidla

2. 1. Digitální teploměr DS18B20

teploměr DS18B20. https://learn.adafruit.com/adafruits-raspberry-pi-lesson-11-ds18b20-temperature-sensing Teploměr DS18B20 vzhledově připomíná tranzistor:



Jeho pin GND spojíme se stejnojmenným pinem RPI, Vdd s pinem 3V3 RPI a DQ s GPIO4. Mezi DQ a Vdd je ještě třeba připojit odpor 4k7. Dále je potřeba provést úpravy v sw (os RPI):
Na konec souboru **/boot/config.txt** dopíšeme řádku

```
dtoverlay=w1-gpio
```

a restartujeme RPI

```
sudo reboot
```

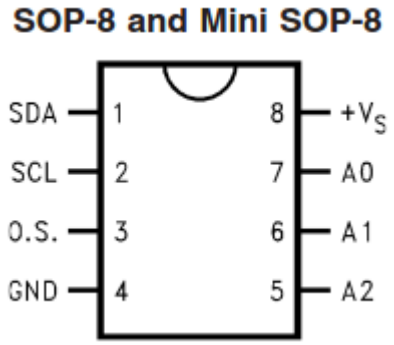
Poté v adresáři devices provedeme následující akce:
sudo modprobe w1-gpio
sudo modprobe w1-therm
cd /sys/bus/w1/devices
ls
cd 28-xxxx

poté ještě zadáme
cat w1_slave

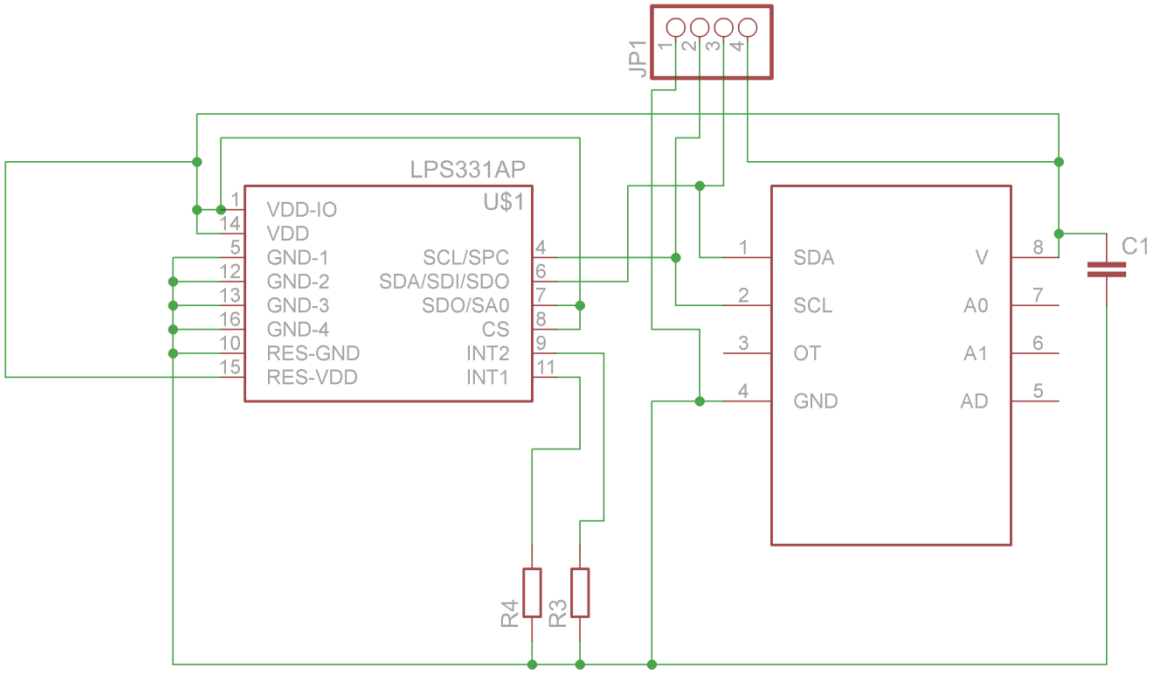
Výsledkem bude výpis registrů následovaný naměřenou hodnotou teploty (po vydělení 1000). Můžeme ale použít i program napsaný např. v Pythonu nebo v C.

2.2. Digitální teploměr LM75

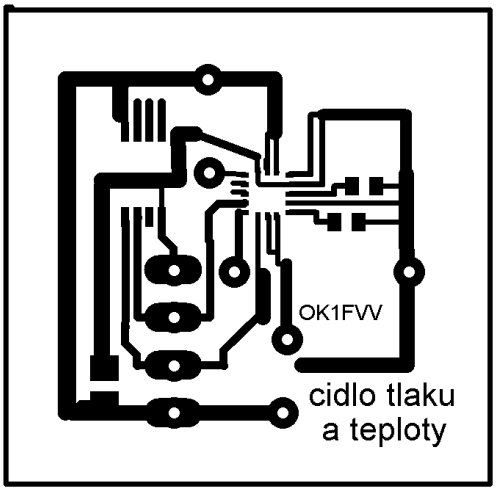
Další možností je využít i2c digitální teploměr. Je jich celá řada a jsou do značné míry kompatibilní s jejich představitelem LM75.

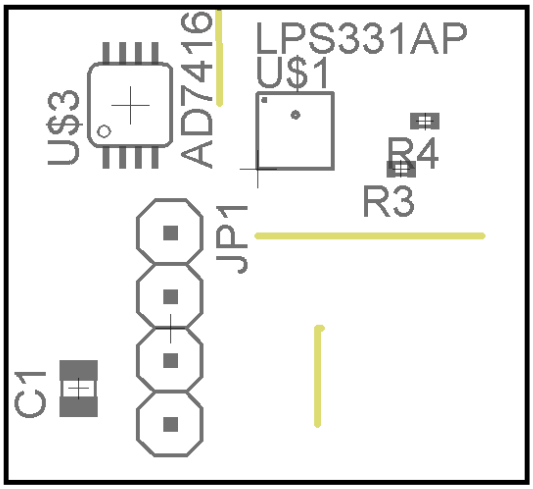


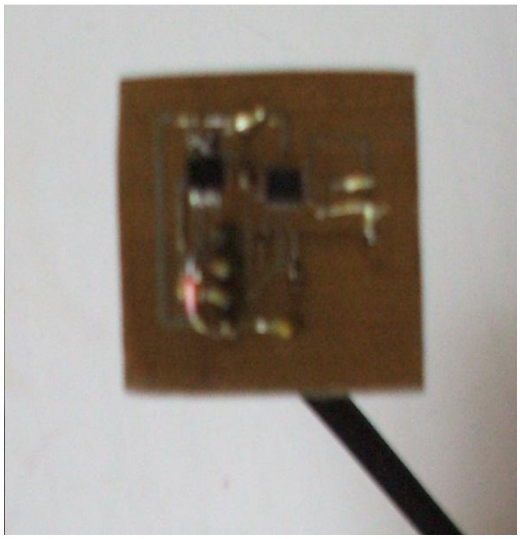
Takovým obvodem je např. AD7416ARMZ, který jsme získali jako „free samples“. Umístili jsme ho na jednu destičku spolu s čidlem tlaku LPS331APYR rovněž komunikujícím pomocí i2c.



Obrazec PCB a rozložení součástí je na následujících obrázcích:





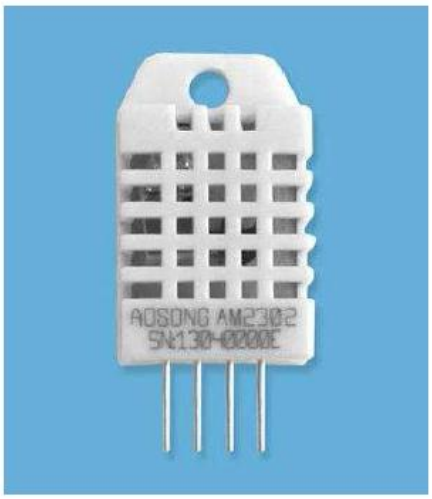


Programovou obsluhu zabezpečí např. knihovna LM75.py. Příklad jejího použití je např.

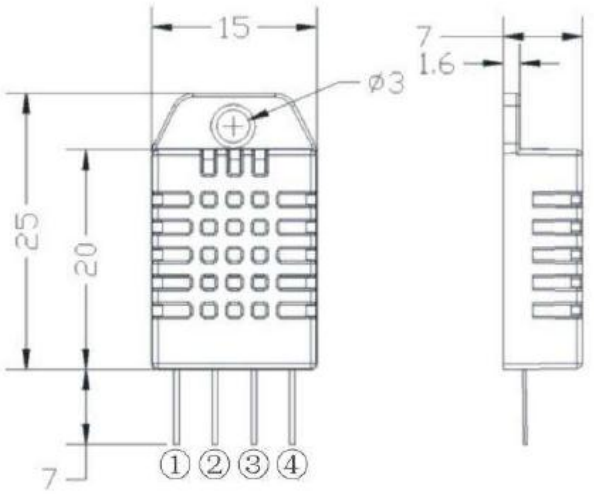
```
import LM75
sensor = LM75.LM75()
print sensor.getTemp()
```

Obdobně programovou obsluhu čidla tlaku LPS331 zajišťuje program ReadSensor.py

2.3 Čidlo vlhkosti DHT11

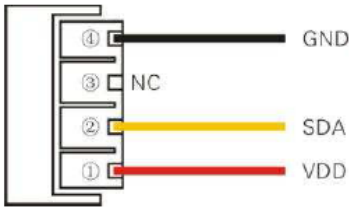


Physical map



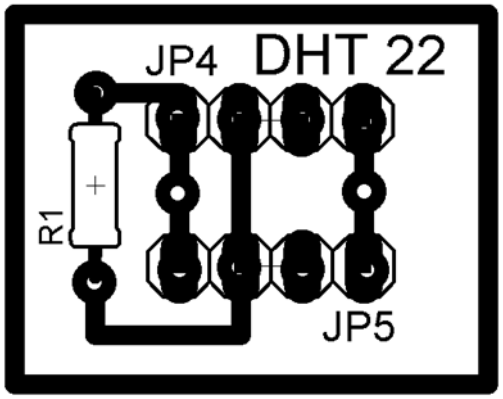
Dimensions (unit: mm)

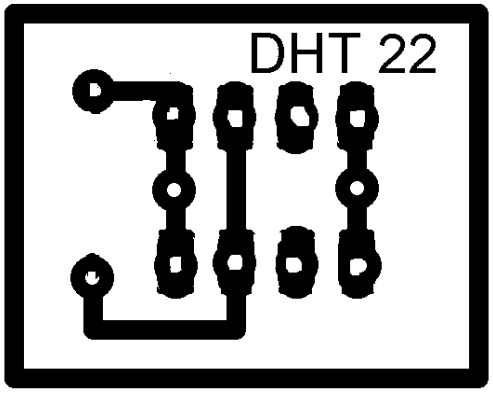
Pin	Name	Description
①	VDD	Power (3.3V–5.5V)
②	SDA	Serial data, bidirectional port
③	NC	Empty
④	GND	Ground



DH11	Raspberry Pi
VCC	3.3v
GND	GND
DATA	GPIO#4

Mezi VCC a DATA je třeba ještě zapojit odpor 4k7 až 10k.
I pro čidla vlhkosti jsme použili jednoduchou destičku PCB:



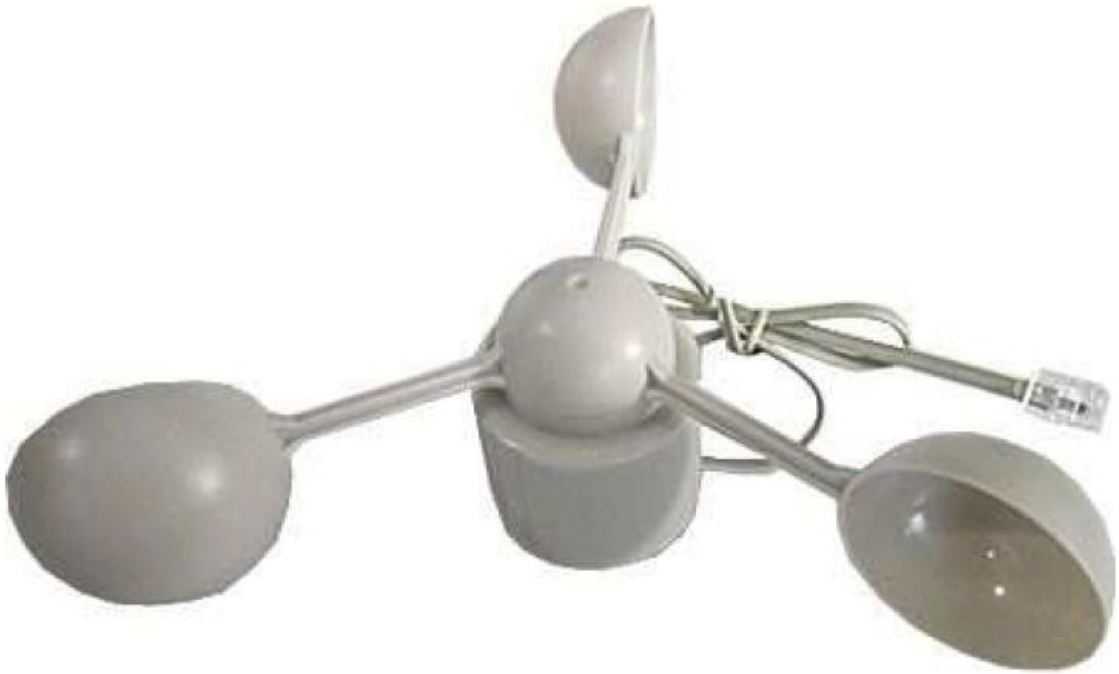


Na http://www.uugear.com/portfolio/dht11-humidity-temperature-sensor-module/ najdeme návod k zprovoznění čidla vlhkosti i obslužný sw.

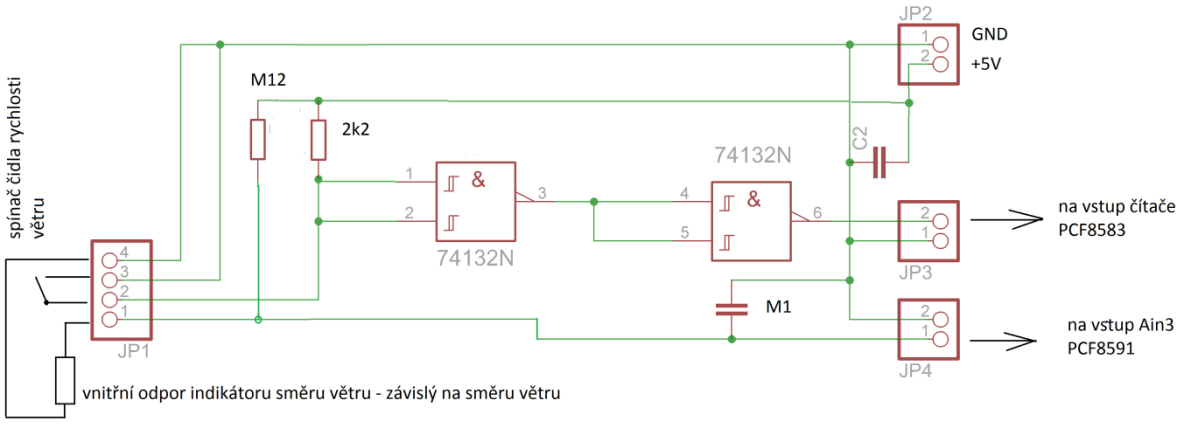
3. Kapitola - Rozšíření meteostanice

V předchozích kapitolách jsme si ukázali, jak pracovat s čidly teploty, tlaku a vlhkosti. To by pro jednoduchou meteostanici mohlo stačit. Můžeme ji však rozšířit o další měření.
Kromě teploty, tlaku a vlhkosti budeme ještě měřit rychlost a směru větru. K tomu využijeme následující dvě čidla: Anemometr T114 a indikátor směru větru T115.

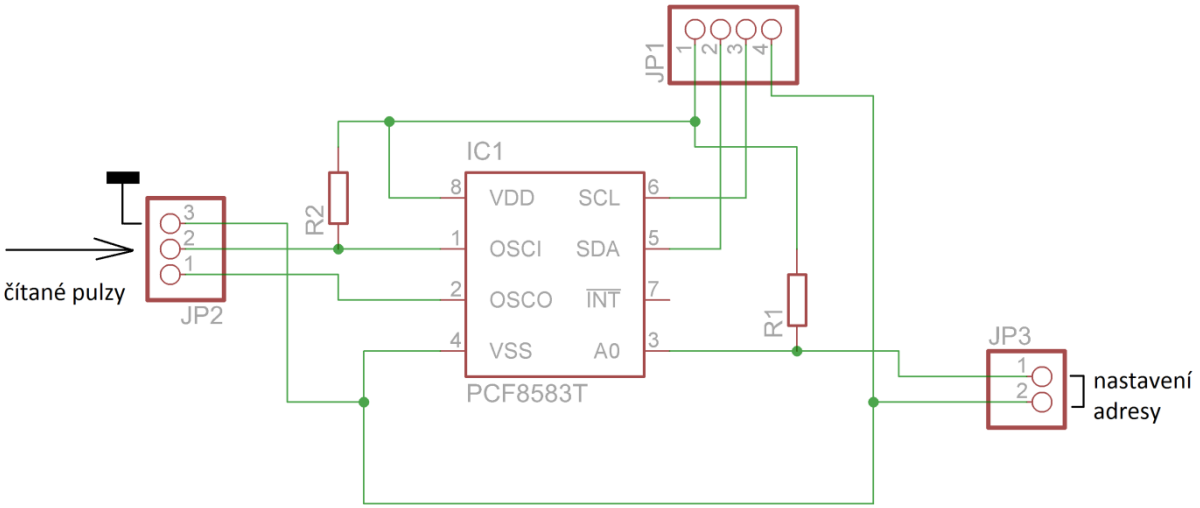
3.1 Anenometr



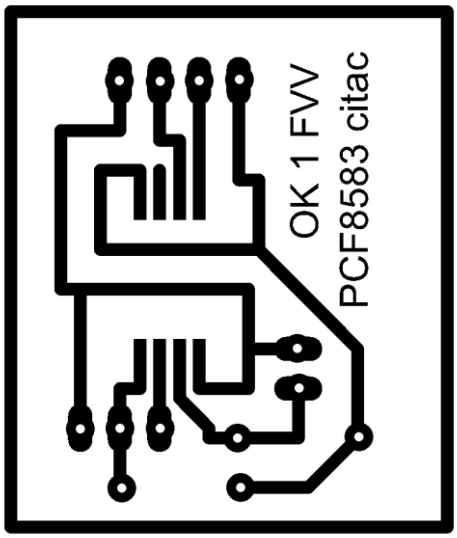
Anemometr T114 sloužící jako čidlo rychlosti větru obsahuje uvnitř spínač, který se sepne 2x během jedné otáčky. Do série s tímto spínačem připojíme odpor 2k2 který je druhým koncem připojen ke zdroji +5V. Paralelně ke spínači ještě připojíme kondenzátor M1 pro potlačení zákmitů při spínání. Při otáčení anenometru tak dostáváme sérii impulzů, které přivedeme na vstup hradla NAND s Schidtovým obvodem 74132. Takto ošetřené pulzy již čítame obvodem s PCF8583. Získáme tak počet pulzů za sekundu. Po vynásobení konstantou dostaneme rychlost větru v m/s nebo km/hod. Na stejnou destičku jsme ještě umístili odpor M12 sloužíci jako předřadný odpor indikátoru směru větru, jehož vnitřní odpor právě závisí na směru větru. S odporem M12 tak tvoří odporový dělič, na němž měříme napětí AD převodníkem PCF8591, čímž získáme informaci o směru větru.

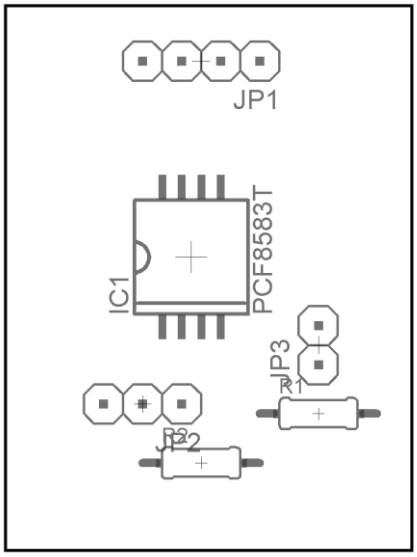


Jako konektor JP1 jsme použili RJ11, neboť ten využívá i anenometr T114 spolu s větrnou růžicí T115.
Schéma zapojení čítače:

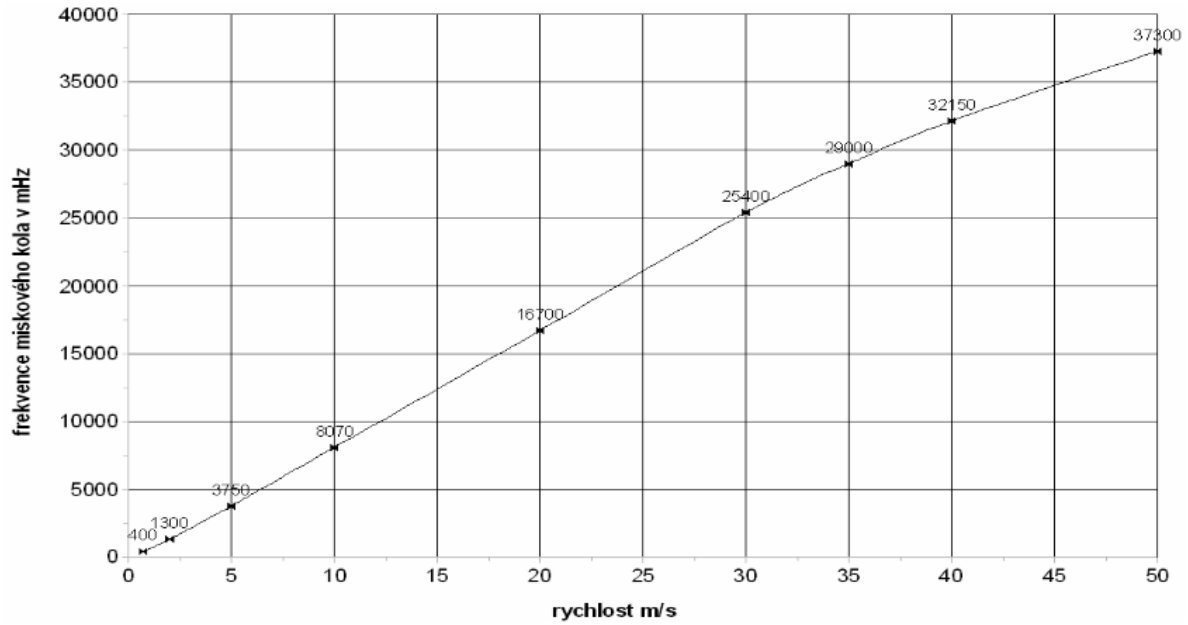


deska spojů PCB a rozložení součástí:





Zbývá určit konstantu, kterou vynásobíme počet pulzů za sec, abychom dostali rychlost větru. Na http://pihrt.com/elektronika/298-moje-raspberry-pi-plugin-prutokomer je uvedeno 2.4 km/hod na jeden impulz a dále ještě graf

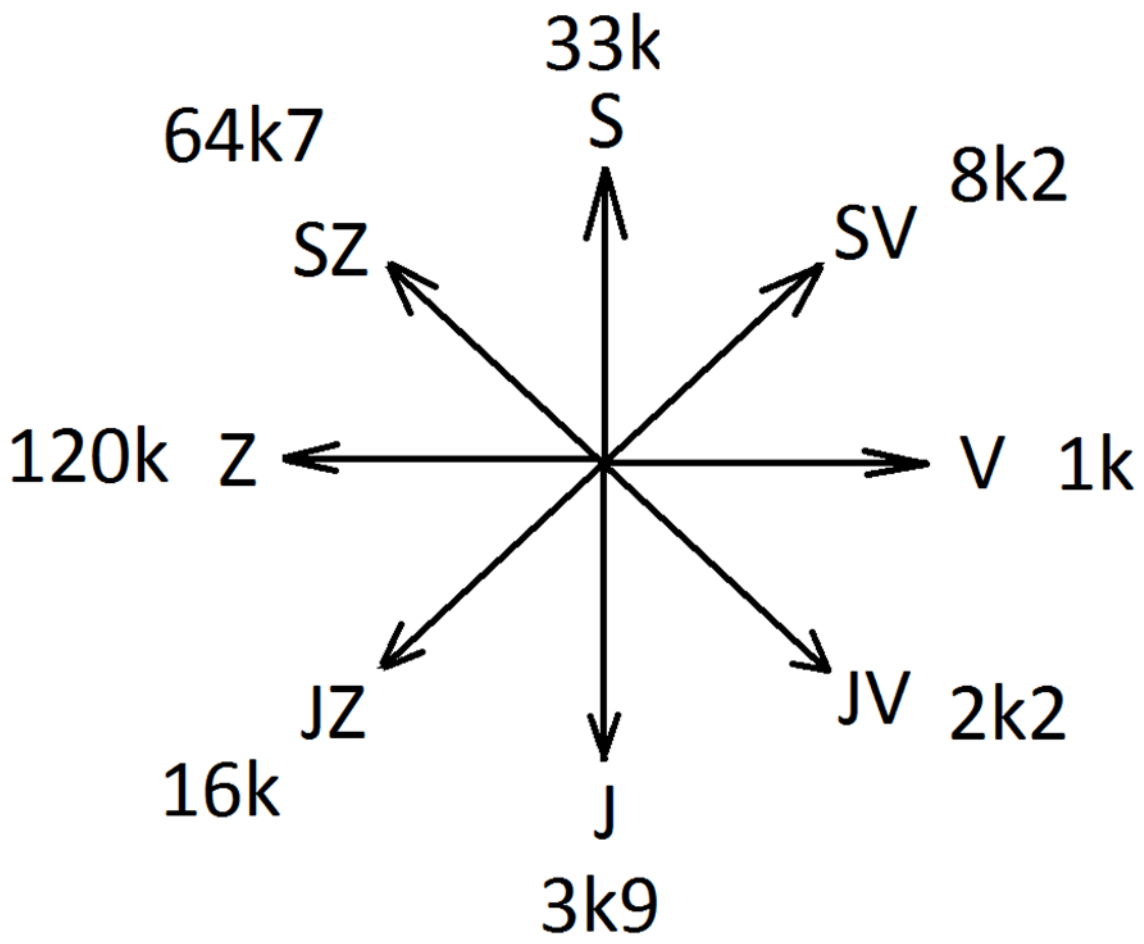


Můžeme se pokusit tento údaj zkontrolovat. Víme totiž, že poloměr růžice je 7cm, tj středy polokoulí při jedné otáčce urazí dráhu 2 * π * R což je přibližně 14 * 3.14 = 44 cm tj. 0.44m Předpokládám-li, že se růžice otáčí stejnou rychlostí jako je rychlost větru, znamená to že při rychlosti 0,44m/s dostaneme 2 impulzy, tj konstanta vychází 0.22m/s na jeden impulz což je 3600 * 0.22 m/hod = 1.584 km/hod

3.2 Větrná růžice

Kromě rychlosti větru potřebujeme určovat i směr větru. K tomu využijeme větrnou růžici viz obr.:

Umí rozlišit 8 směrů S, J, V, Z, SV, SZ, JZ a JV. V závislosti na natočení je sepnut příslušný spínač, která připojí odpovídající odpor ke dvojici výstupních vodičů. Naměřili jsme pro různé směry odpory viz obr.:



Z tohoto odporu a odporu M12 jsem vytvořil odporový dělič připojený k +5V a k témuž napětí jsem připojil i Uref obvodu PCF8591. Vstup 8bitového AD převodníku jsme připojili ke středu odporového děliče. Výstupem osmibitového AD převodníku jsou čísla z intervalu 0 až 255. V našem případě dostáváme pro jednotlivé směry následující čísla:

V	JV	J	SV	JZ	S	SZ	Z
2	4	8	16	29	52	82	113

Úspěšně jsme odzkoušeli následující kód

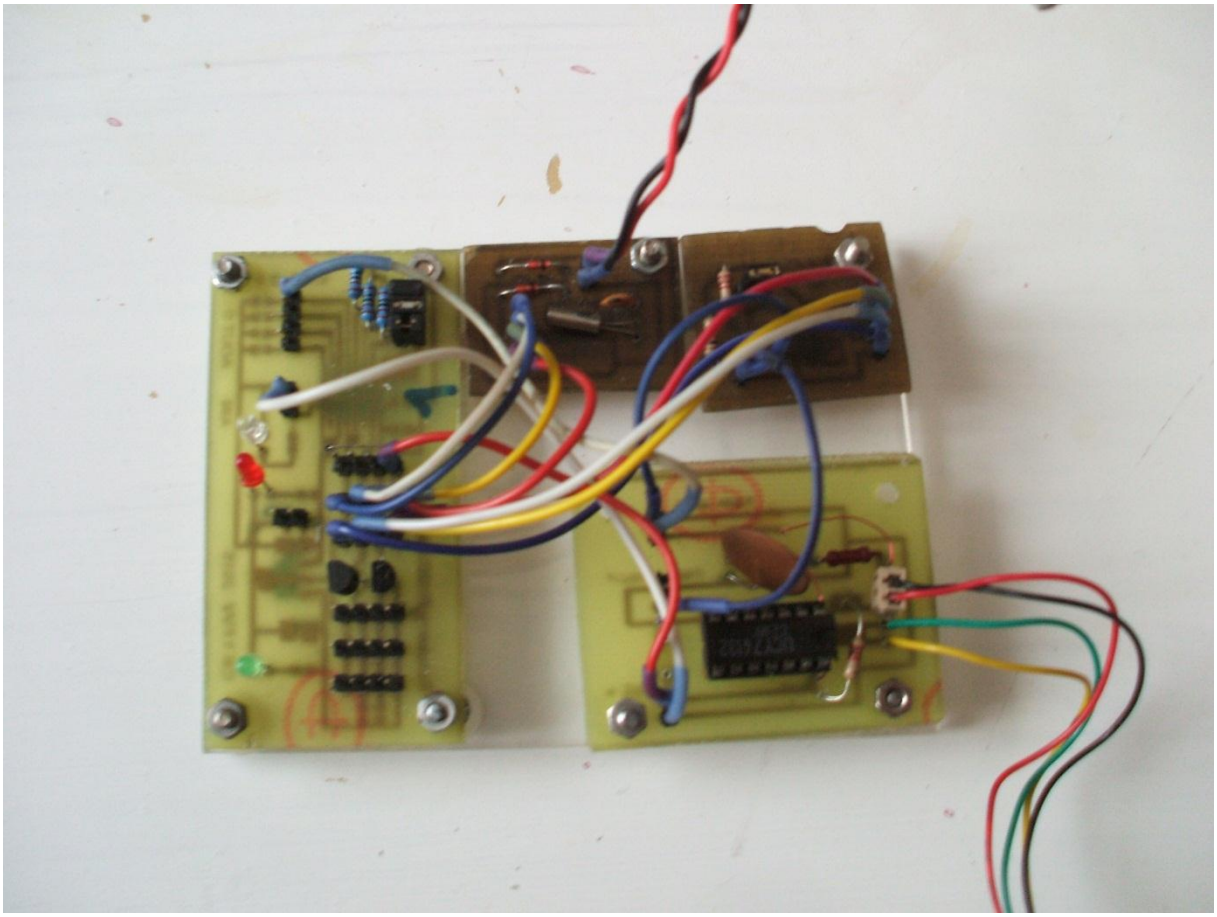
```
#!/usr/bin/python
import smbus
import time
#  meric smeru vetru / vetrna korouhvicka T115  / pripojuje se k AD PCF8591  No2
# I2C sbernice novejsi RPI=1 starsi RPI=0
ADC = smbus.SMBus(1)

def get_volt(data):
```

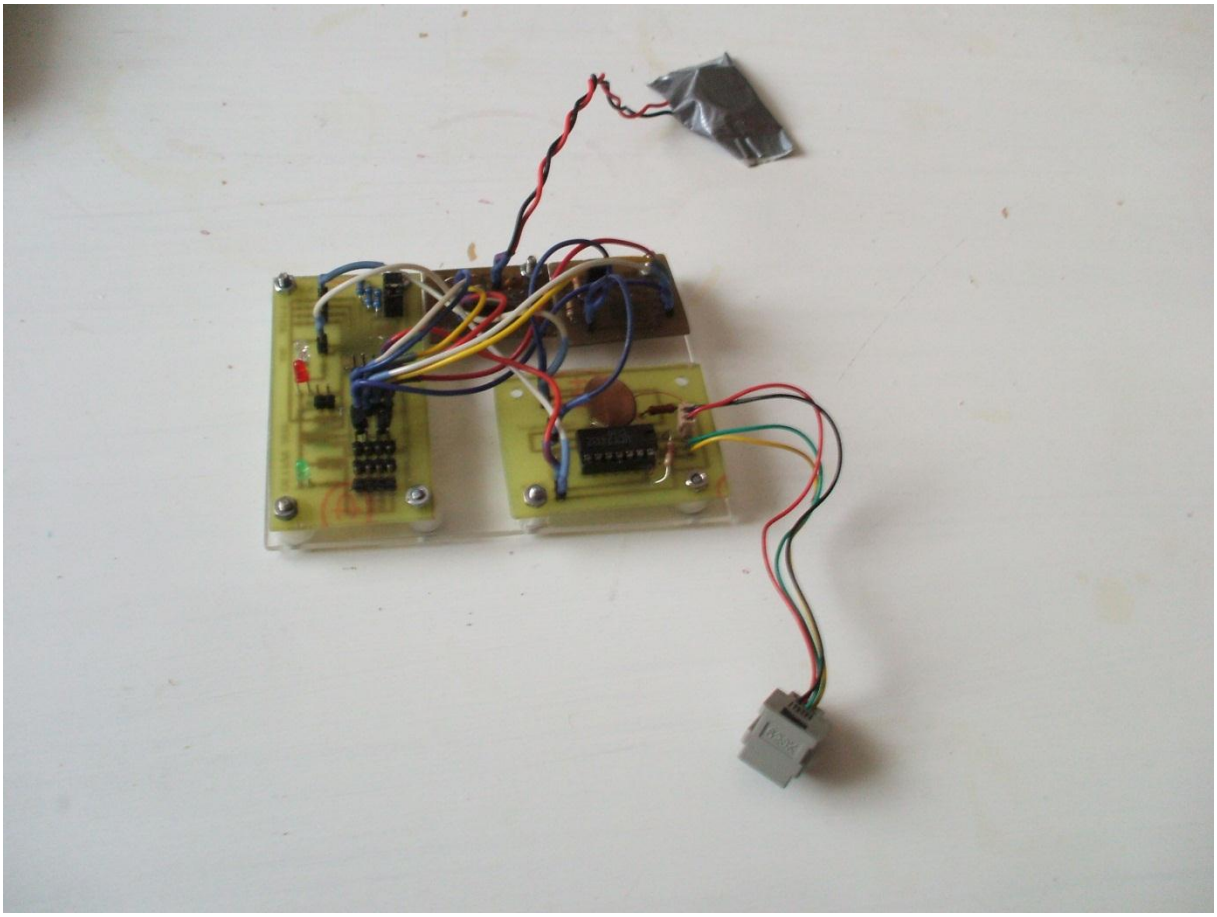
```
    """Return voltage 0-5.0V from number"""
    volt = (data*5.0)/255
    volt = round(volt,1)
    return volt

def get_temp(data):
    """Return temperature 0-100C from data"""
    temp = ((data*5.0)/255)*100.0
    temp = round(temp,1)
    return temp

def get_now_measure(AD_pin):
    """Return number 0-255 from A/D PCF8591"""
    try:
      ADC.write_byte_data(0x48, (0x40 + AD_pin), AD_pin)
      return ADC.read_byte(0x48)
    except:
      return 0

def get_write_DA(Y):  # PCF8591 D/A converter Y=(0-255) for future use
    """Write analog voltage to output"""
    try:
      ADC.write_byte_data(0x48, 0x40, Y)
    except:
      return


while True:  # Smycka
    ad0 = get_now_measure(1)
    ad1 = get_now_measure(2)
    ad2 = get_now_measure(3)
    ad3 = get_now_measure(4)
#   print ad0,ad1,ad2,ad3
#    print ad3

    if ad3 <= 3:
      print "V - vychodni vitr"
    if 3 < ad3 <= 6:
      print "JV - jihovychodni vitr"
    if 6 < ad3 <= 12:
      print "J - jizni vitr"
    if 12 < ad3 <= 24:
      print "SV - severovychodni vitr"
    if 24 < ad3 <= 40:
      print "JZ - jihozapadni vitr"
    if 40 < ad3 <= 59:
      print "S - severni vitr"
    if 59 < ad3 <= 89:
      print "SZ - severozapadni vitr"
    if 89 < ad3:
```

```
    print "Z - zapadni vitr"
time.sleep(1)
```

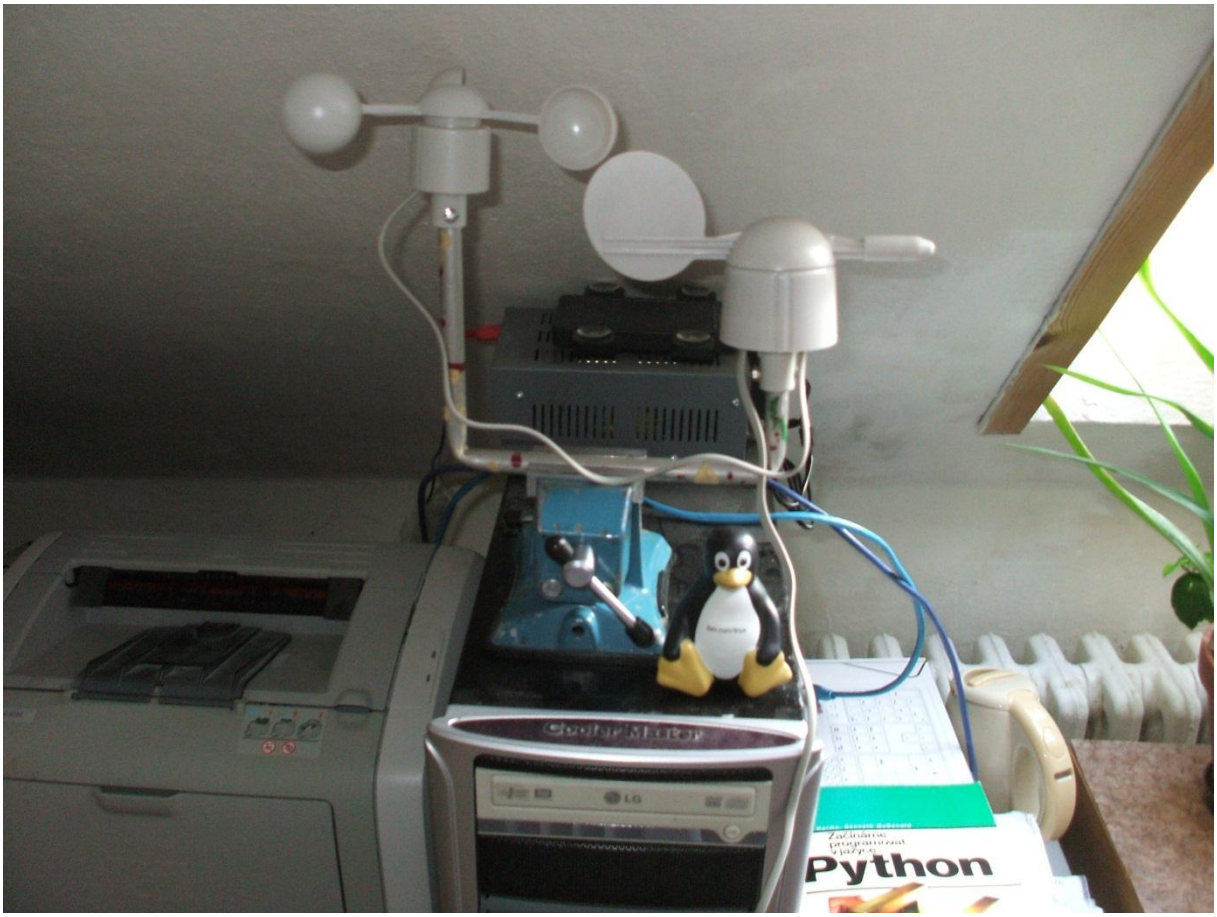
3.3 konstrukční provedení

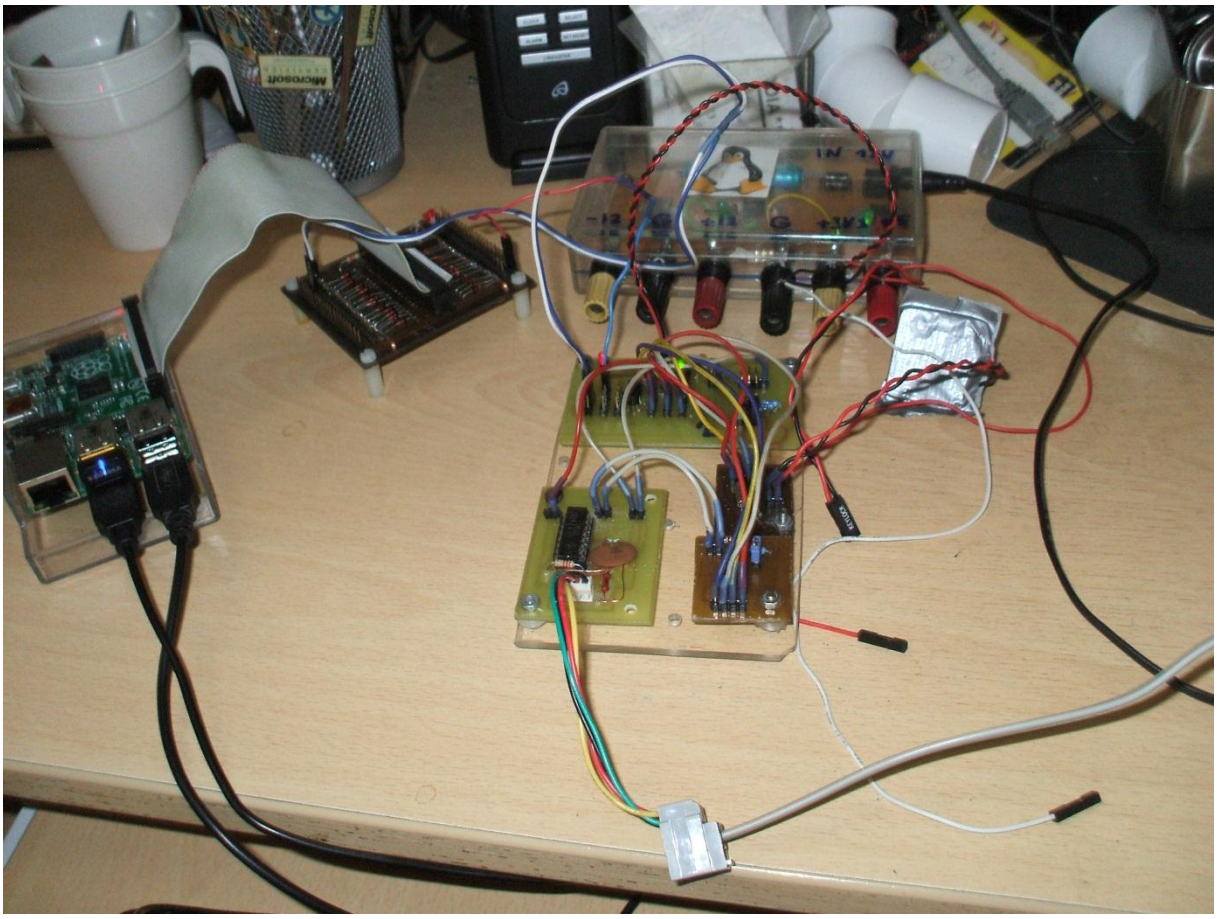
Veškeré i2c bloky meteostanice a interface k čidllům měřícím rychlost a směr větru jsme přišroubovali k základní destičce z plexiskla a bloky jsme propojili vodiči připájenými cínem ke špičkám příslušných konektorů na destičkách:





Nakonec ještě několik fotografií z pracoviště při vývoji meteostanice:


Python



Závěr

Naším úkolem bylo navrhnout a zrealizovat meteostanici. Naší první verzí se nám podařilo splnit požadavky zadání, nicméně je zde stále možnost dalšího vylepšování systému na základě jeho testování, kalibrace čidel a zejména vlastního zpracování dat a jejich interpretace.