Členové týmu

- Vít Jánoš (zodpovědný za PWM)
- Vojtěch Kudela (zodpovědný za obsluhu a správu GitHub)
- Jakub Kupčík (zodpovědný za získání dat)
- Antonín Putala (zodpovědný za GUI a UART)

Teoretický popis a vysvětlení

V době prudkého nárůstu světové populace jsou hledány způsoby, jak zvýšit světovou produkci potravin. Jednou z možností, která se nabízí je pěstování ve **sklenících**. Velkou výhodou skleníků je, že potravinová produkce v nich není vázána na vegetační cyklus ani na klimatické podmínky. Podmínky uvnitř skleníku je možné udržovat systémem senzorů a akčních členů, které zajistí **optimální klima** pro zde pěstované rostliny.

Jak známo rostliny ke svému životu potřebují [10]:

1. vodu a živiny,
2. vzduch a půdu,
3. světlo a teplo,
4. prostor a čas.

Půdu a živiny je nutné zajistit při setí nebo sadbě. Vzduch bude zajištěn přístupem čerstvého vzduchu. Požadavek na prostor je omezující, co se týče rostlin, které jsme schopni na ploše vysadit a čas bohužel regulovat nelze, je tedy nezbytné nechat rostlinu růst do doby, než ponese plody.

Naopak je možné regulovat **teplotu** okolí, **světlo** a vodu, která se projeví jako **půdní vlhkost**. Tento projekt se zaměřuje na pěstování **tropických rostlin**. Je potřeba myslet na to, že tropické rostliny rostou ve velmi vlhkém prostředí, proto je nutné regulovat také **vlhkost vzduchu** [11].

Pro otestování možností bylo realizováno zařízení schopné, jak **měřit**, tak i **regulovat** zvolené veličiny v skleníku. Tyto veličiny je pochopitelně nutné v čase měnit, aby pokud možno odrážely, denní cykly a co nejlépe odrážely klima, ve kterém rostlina přirozeně roste. Protože jako pěstitel nemáme možnost neustále sledovat vývoj veličin, naměřené **hodnoty veličin** je vhodné **ukládat** pro další zpracování. Celé zařízení bylo naprogramováno v rozhraní Platform.io pro mikrokontroler ATMEGA328P. Debugování a testování bylo provedeno na desce Arduino UNO.

Byla snaha, aby obsluha zařízení byla uživatelsky přívětivá. Proto namísto použití tlačítek, nebo maticové klávesnice bylo zvoleno **ovládání prostřednictvím osobního počítače**, k čemuž slouží grafické uživatelské rozhraní (GUI). Mezi velké výhody tohoto řešení patří velká variabilita funkcí a snadná adaptovatelnost pro budoucí modifikace. Toto rozhraní mimo jiné umožňuje zpětně zobrazit naměřené hodnoty pozorovaných veličin.

Hardwarový popis

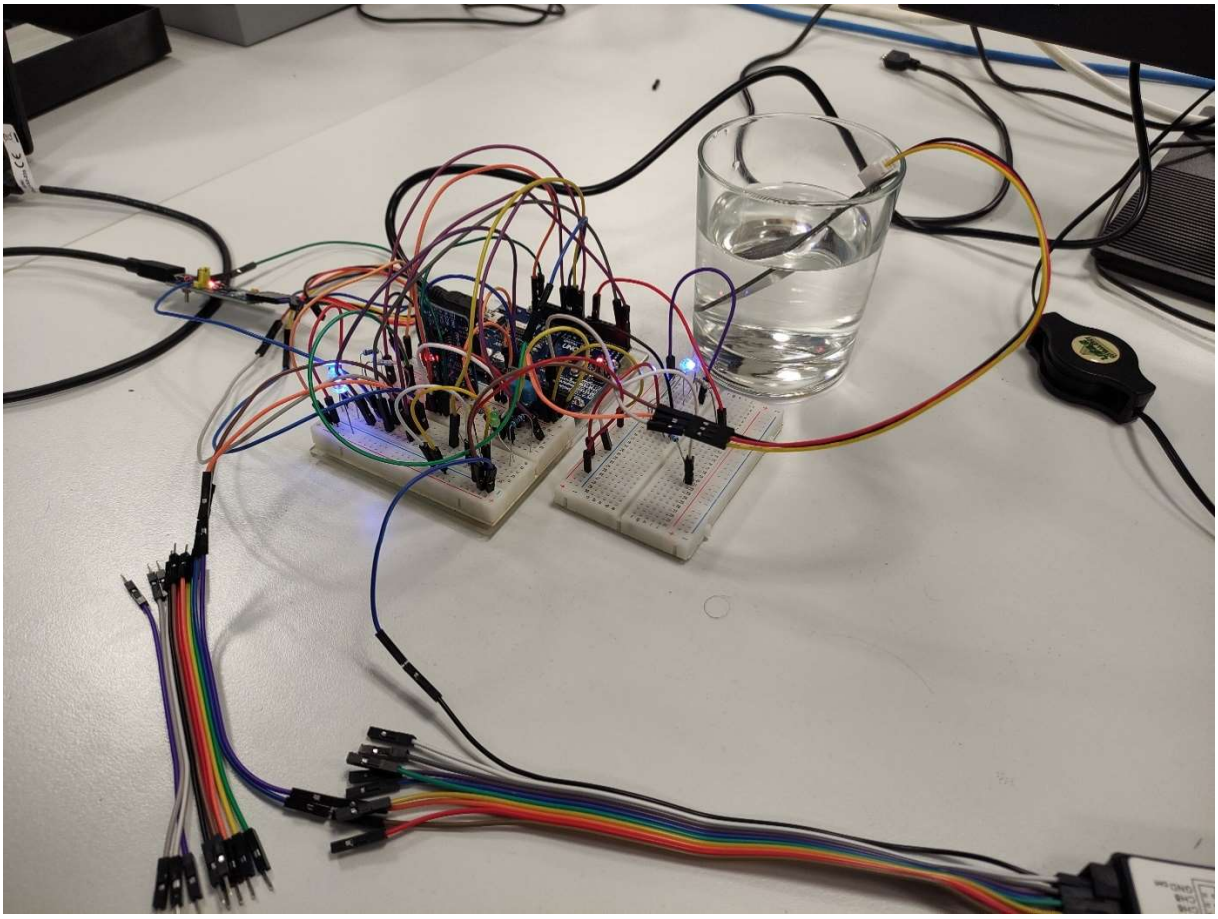
Zařízení představuje prototyp měřicího a řídicího členu pro tropický skleník. Umožňuje měření teploty, osvětlení, půdní vlhkosti a vlhkosti vzduchu. Tyto hodnoty jsou odečítány v reálném čase. Odečítání hodnot veličin zajišťují tyto senzory:

1. DHT12 – kombinované teplotní a vlhkostní čidlo,
2. Kapacitní senzor půdní vlhkosti v1.2 – Arduino,
3. Fotorezistor.

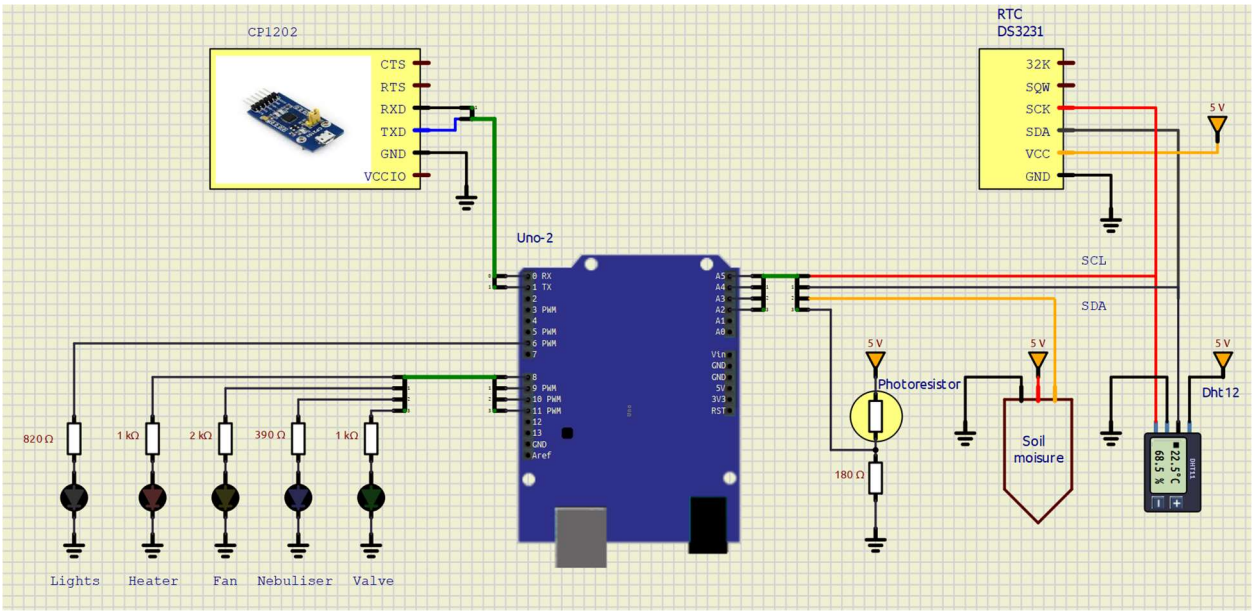
Aby odečítané hodnoty byly v reálném čase, byly použity hodiny reálného času RTC DS3213.

Další funkcí je regulace těchto veličin. K tomu slouží výstupní periférie. V tom to případě byl uvažován **topný člen**, **ventilátor**, **nebulizér**, **ventil** pro ovládání hadice, která bude zajišťovat závlahu a LED pásek, který bude představovat osvětlení. Regulaci teploty zajišťuje topný člen a ventilátor a regulaci vlhkosti vzduchu zajišťuje nebulizér a ventilátor.Pro otestování funkce byly tyto periferie nahrazeny různobarevný **LED** diodami.

Schéma + fotka testovacího zařízení



Obr. 1 Prototyp měřicího a regulačního zařízení pro skleník

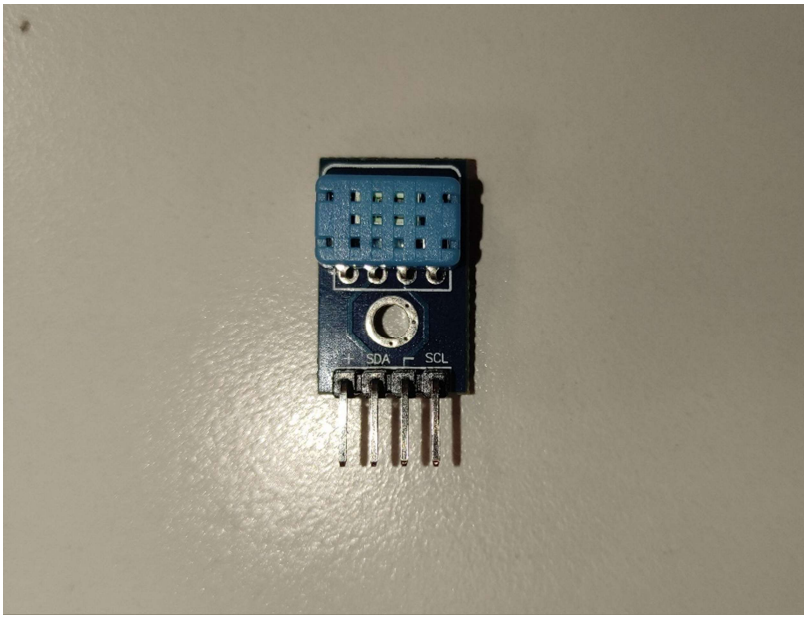


Obr. 2 Schéma zapojení

Periferie

Následující odstavce popisují použité periférie a jejich zapojení.

Kombinované teplotní a vlhkostní čidlo



Obr. 3 Kombinované teplotní a vlhkostní čidlo

Pro měření teploty i vlhkosti vzduchu bylo použito kombinované čidlo DHT12. Velkou výhodou tohoto čidla je zabudovaný AD převodník a komunikace pomocí I2C. Toto čidlo tedy zabere na desce pouze dva piny a je možné připojit množství dalších periférii, aniž by rostly požadavky na počet pinů desky. Vodiče I2C je nutné připojit k napájecímu napětí přes pull-up rezistory. Jsou použity vnitřní pull-up rezistory mikrokontroleru.

Pro správnou funkci je nezbytné připojit pin SCL na port PC5 a pin SDA na port PC4. Čidlo je také nutné napájet 5 V. Z tohoto čidla je pouze čteno. Zápis probíhá pouze za účelem nastavení místa paměti, ze kterého bude čteno. Toto čidlo, podobně jak je to běžné u jiných zařízení typu Slave, je že čítač paměti je automaticky inkrementován po každém čtení. Je tedy možné během jediného čtení přečíst veškerý obsah paměti čidla.

Tab. 1 Paměť čidla DHT12

Adresa	0x00	0x01	0x02	0x03	0x04
Obsah paměti	Vlhkost – celá část	Vlhkost – desetinná část	Teplota – celá část	Teplota – desetinná část	Kumulativní součet

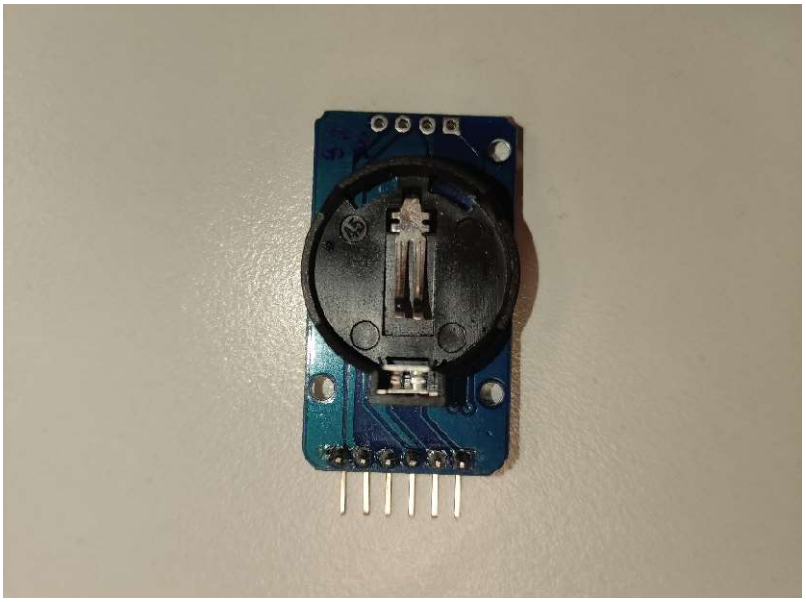
Protože operace s desetinnými čísly, především s plovoucí desetinou čárkou, jsou pro mikrokontroler velmi náročné, jsou změřené hodnoty v paměti uloženy zvlášť desetinná, zvlášť celá část. Kumulativní součet slouží ke kontrole přenesených dat. Platí:

$CS = H_i + H_d + T_i + T_d$.

Jednotlivé veličiny jsou vypsány dle pořadí v tabulce.

Neméně důležitou informací je I2C adresa čidla, která umožňuje mikrokontroleru (Masteru) komunikovat přímo s tímto čidle. Čidlo má adresu 0x5c.

Hodiny reálného času



Obr. 4 Hodiny reálného času

Hodiny reálného času řeší problém časování. Naměřeným hodnotám veličin je přiřazen čas. Data jsou vysílána přibližně každou sekundu. Ovšem program je napsán v jazyce C nikoli v jazyku symbolických adres a zařízení není optimalizováno, aby přesně každou sekundu poslalo zprávu. Z tohoto důvodu není možné tento signál použít k synchronizaci dat. Další nespornou výhodou je, že hodiny reálného času podávají informaci také o dnu, měsíci a roku, což velmi zjednodušuje následné zpracování údaje o času.

Použité hodiny reálného času DS3213 umožňují stejně jako teplotní a vlhkostní čidlo komunikovat přes I2C. Připojení je obdobné jako v případě teplotně-vlhkostního čidla. V tomto případě je nutné, jak data z hodin číst, tak i zapisovat, aby bylo možné nastavit aktuální čas.

Pro čtení i zapisování je nutné znát strukturu paměti, která je dána v následující tabulce:

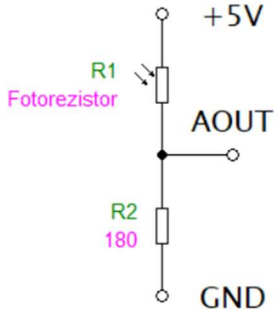
Tab. 2 Struktura paměti DS3213

Adresa	Bit							
	7	6	5	4	3	2	1	0
0x00	0	Desítky sekund			Sekundy			
0x01	0	Desítky minut			Minuty			
0x02	0	0	Desítky hodin		Hodiny			
0x03	0	0	0	0	0	Den v týdnu		
0x04	0	0	Desítky dnů v týdnu		Den v měsíci			
0x05	1	0	0	Desítky měsíců	Měsíce			
0x06	Desítky let				Roky			

Takto nastavené RTC bude čítat v 24 hodinovém formátu. Je nastaveno 21.století. I2C adresa hodin je 0x68.

Čidlo osvětlení

Čidlo osvětlení je realizováno pomocí děliče, přičemž odpor, na kterém není měřeno napětí je nahrazen fotorezistorem. Tato konfigurace byla zvolena, aby s rostoucím osvětlením rostlo napětí na výstupu AOUT. Protože je výstup analogový bylo toto čidlo připojeno k analogovému vstupu. Konkrétně byl zvolen PC2.



Obr. 6 Schéma zapojení čidla osvětlení

Obr. 5 Fotorezistor

Tato konfigurace není sama o sobě schopna dát smysluplnou hodnotu osvětlení, je nutný přepočet. Byla změřena závislost výstupního napětí vyjádřeného 10-bitový číslem na osvětlení. Měření probíhalo pouze pro tmu a maximální osvětlení, což bylo provedeno tak, že dioda svítila na plný výkon do fotorezistoru. Tyto dva body byly proloženy přímkou. Samozřejmě se jedná o aproximaci, ve skutečnosti je závislost nelineární. Nicméně pro orientační zjištění hodnoty osvětlení a přibližné nastavení spínacího prahu je taková aproximace zcela dostačující.

Tab. 3 Hodnoty pro aproximaci převodní charakteristiky čidla osvětlení

N [–]	164	480
E [lx]	16	358

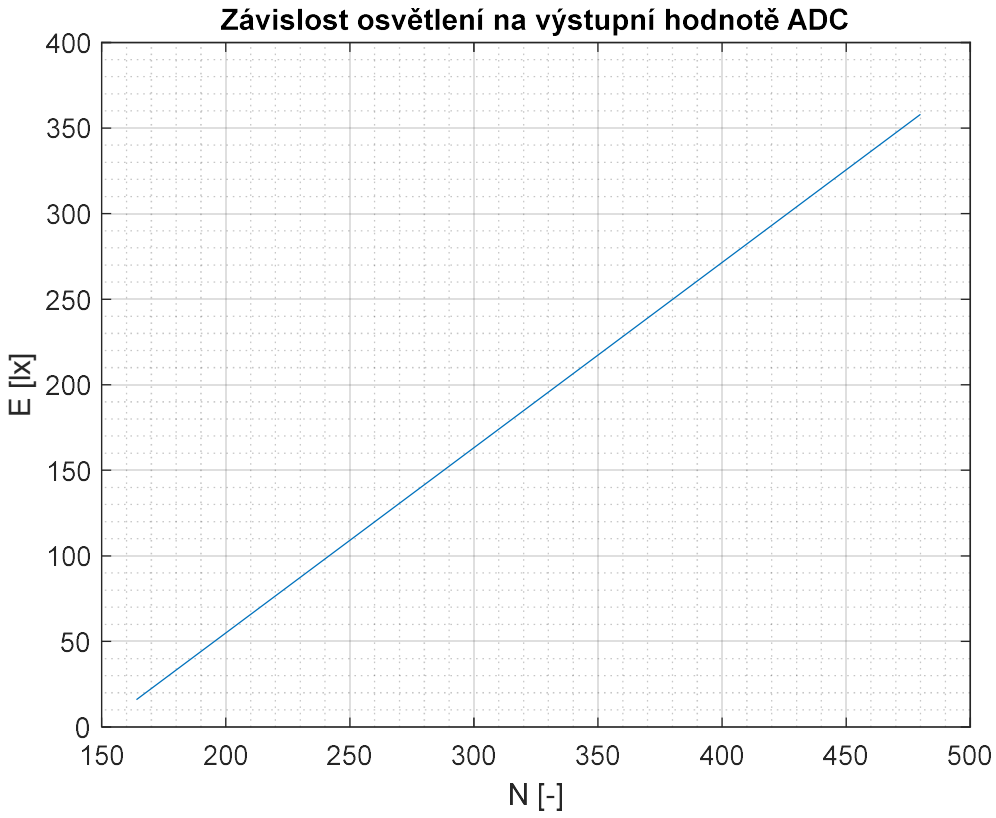
Převodní charakteristika byla aproximována následující funkcí:

$$\Delta f = \frac{E_2 - E_1}{N_2 - N_1} = \frac{358 - 16}{480 - 164} = 1 \text{ lx}$$

$$f_0 = E_1 = 16 \text{ lx}$$

$$x_0 = N_1 = 164 \text{ [–]}$$

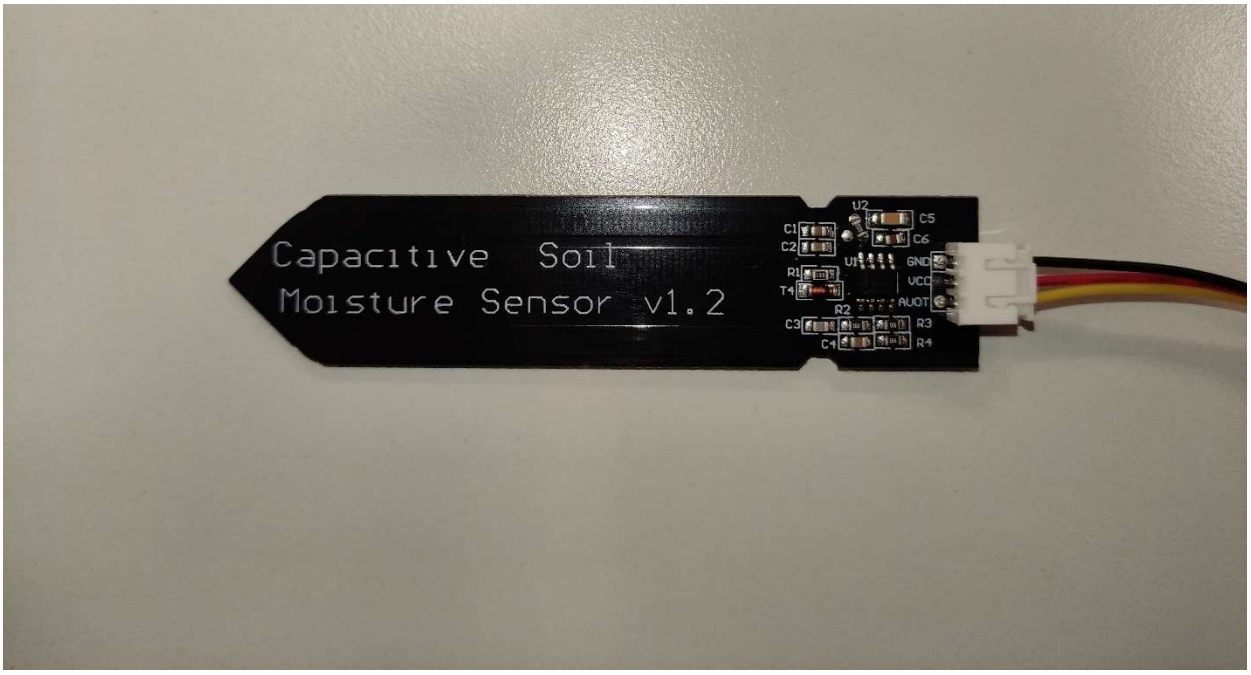
$$E(N) = \Delta f \cdot (N - x_0) + f_0 = (N - 16) + 164$$



Obr. 7 Závislost zobrazené hodnoty osvětlení na výstupní hodnotě ADC

Zvolená maximální hodnota osvětlení působí poměrně zvláštně. Nicméně tato hodnota byla zvolena pouze proto, aby nebylo nutné řešit operace s desetinnými čísly, neboť tyto operace jsou pro mikrokontroler podstatně náročnější.

Čidlo půdní vlhkosti



Obr. 8 Čidlo půdní vlhkosti

Měření půdní vlhkosti je realizováno pomocí kapacitního čidla půdní vlhkosti z Arduina. Toto čidlo má napájení 5 V, zemi a analogový výstup. Ten byl připojen na analogový pin PC3. K tomuto čidlu bohužel není k dostání převodní charakteristika, a tak byla určena experimentálně a aproximována přímkou. Jako minimální vlhkost byl uvažován stav, když bylo čidlo na vzduchu. Maximální vlhkost byla uvažována, pokud bylo čidlo umístěno ve sklenici vody.

Tab. 4 Hodnoty pro aproximaci převodní charakteristiky čidla půdní vlhkosti

N [–]	233	186
SM [%]	0	100

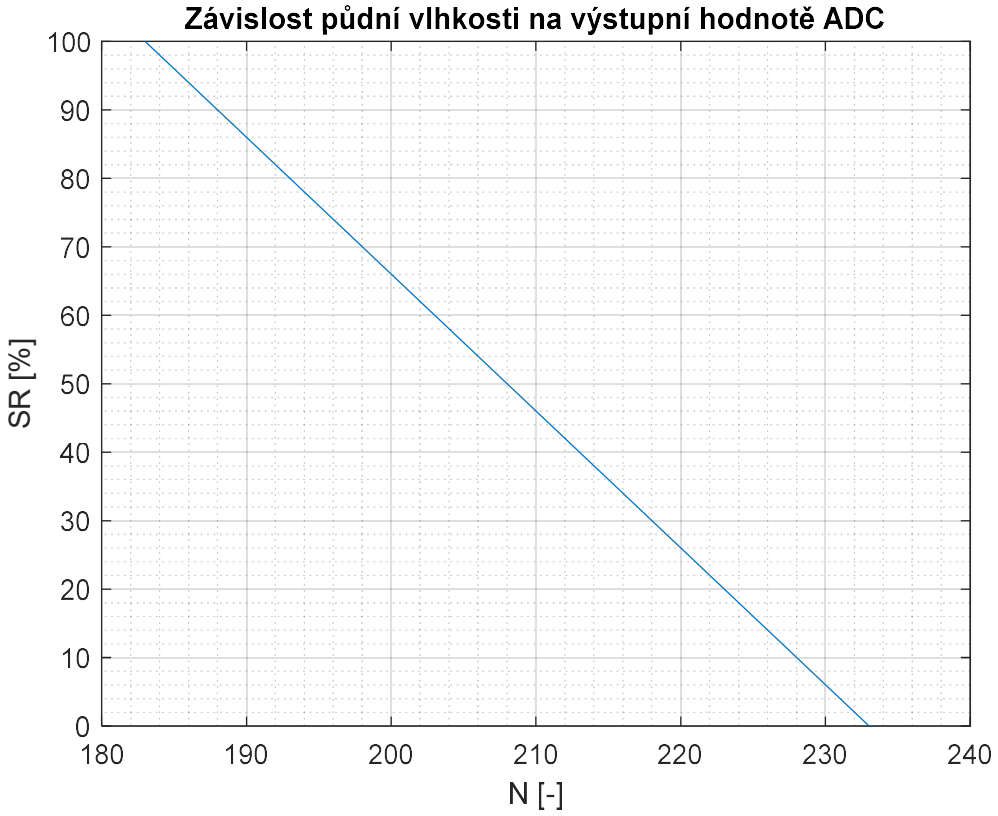
Převodní charakteristika byla aproximována následující funkcí:

$$\Delta f = \frac{SM_2 - SM_1}{N_2 - N_1} = \frac{0 - 100}{233 - 186} = -2\ \%$$

$$f_0 = SM_1 = \ 0\ \%$$

$$x_0 = N_1 = 233\ [-]$$

$$SM(N) = \Delta f \cdot (N - x_0) + f_0 = -2 \cdot (N - 233) = 100 - 2 \cdot (N - 183)$$



Obr. 9 Závislost zobrazené hodnoty půdní vlhkosti na výstupní hodnotě ADC

Stanovený přepočet je opět pouze orientační. V žádném případě se nejedná o přesnou fyzikální hodnotu.

Topná člen, ventilátor, nebulizér a ventil

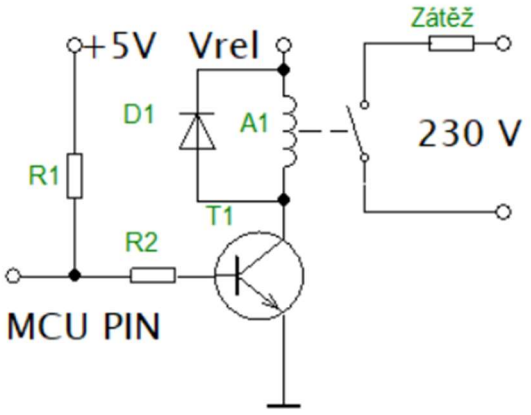
Při regulaci obou vlhkostí a teploty lze s výhodou využít setrvačnosti prostředí, samotný prostor skleníku bude udržovat teplotu i vlhkost. Z toho důvodu přesná PWM regulace nemá význam, postačujícím řešením je **on/off regulace**. Jednotlivé periférie budou připojeny na následující piny:

- PB0 – topná člen,
- PB1 – ventilátor,
- PB2 – nebulizér,
- PB3 – ventil pro zavlažování.

Protože použité periférie vyžadují poměrně vysoký spínací výkon je vhodné použít následující konfiguraci. Mikrokontroler spíná tranzistor a ten spíná relé, které umožňuje spínat velké výkony. Pro tuto aplikaci možné využít bipolární tranzistor. Je nutné uvažovat, jak velký proud bude tranzistor spínat, aby bylo možné správně tranzistor výkonově dimenzovat. Pokud by byl použit NMOS, byl by vhodný driver, protože tyto tranzistory vyžadují vyšší hodnotu spínacího napětí, než je schopen mikrokontroler poskytnout.

Tranzistor je nutné budit dostatečným proudem, proto je na bázi připojen pull-up rezistor. V některých případech lze použít vnitřní pull-up mikrokontroleru. Dále je nutné omezit proud do báze bázovým rezistorem.

Relé je pro tuto aplikaci dostatečně rychlé, problém ale může způsobit, že po určitém počtu sepnutí relé odejde, změna může v nejhorším případě nastat každou sekundu. Abychom bylo této situaci zabráněno, je vždy nastavena hystereze regulované veličiny. Pokud by velký počet spínacích cyklů byl prioritou, stálo by za to zvážit použití optočlenu, nebo SSR relé. Nepříjemnou vlastností relé, je velký záporný napěťový překmit po rozepnutí. Tento překmit způsobí průraz spínacího tranzistoru. Řešením je například **dioda** zapojená paralelně s cívkou relé.



Obr. 10 Doporučené připojení výkonových zařízení

Toto zapojení je **aktivní ve vysoké úrovni**. Pro testování tyto periférie nebyly k dispozici, byly modelovány pomocí diod, které byly tedy též zapojeny, aby byly aktivní ve vysoké úrovni.

Byly použity tyto barvy:

1. Červená – topná člen,
2. Oranžová – ventilátor,
3. Zelená – ventil,
4. Modrá – nebulizér.

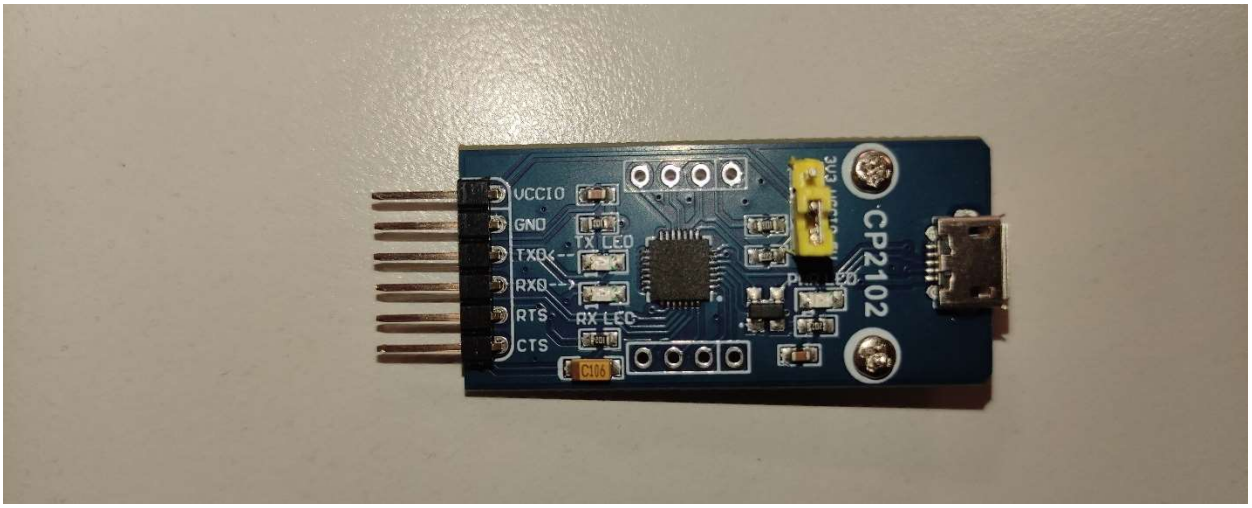
LED pásek

Poněkud těžší úkol představuje regulace osvětlení. Pokud chceme nastavit osvětlení, je nutné nastavit přímo regulovat intenzitu světelného zdroje, protože v tomto případě se neuplatní setrvačnost prostředí. V tomto případě se nabízí použití **PWM regulace**, protože tuto funkci mikrokontroler umožňuje.

LED pásek je nutné připojit k portu PD6. Bylo uvažováno zapojení **active-high**. Je potřeba zvážit odběr LED pásku. Napájení kratších rozměrů zajistí port mikrokontroleru, pro větší odběru je vhodné použít spínací tranzistor. Pro účely testování byl LED pásek nahrazen modrou LED diodou.

Komentář: obsluhu zajišťuje PC

Komunikaci s PC je realizována prostřednictvím sériového portu. Převod z USB na UART zajišťuje převodník CP2102.



Obr. 11 Převodník CP2102

Počítač je na převodník připojen prostřednictvím microUSB. Pro zajištění komunikace s deskou je nutné propiji země převodníku a desky. Dále musí být propojeny datové piny. Vysílací pin TXD převodníku bude připojen na přijímací pin desky RX, zatímco přijímací pin RXD bude připojen na vysílací pin desky TX. Na převodníku byl jumper umístěn tak, aby výstupní napětí na UARTu bylo 5 V.

Pro komunikaci s deskou je nutné používat aplikaci Tropical plants, která obsahuje grafické rozhraní uzpůsobené řízení skleníku.

Softwarový popis

Pro funkci zařízení bylo nezbytné vytvořit jak program pro mikrokontroler, tak pro osobní počítač. Zatímco program pro mikrokontroler byl napsán v jazyce C, program pro osobní počítač byl napsán v Pythonu.

Python

Úkolem programu pro osobní počítač je sbírat data vysílaná zařízením a vysílat instrukce k regulaci měřených veličin. Aplikace též umožňuje změnit čas zařízení a graficky zobrazit změřené hodnoty. Samotnou aplikaci spustí soubor Tropical_plants. Tento program vytvoří aplikaci GUI_main a spustí ji.

Podrobný popis všech použitých tříd a funkcí se nachází v této dokumentaci.

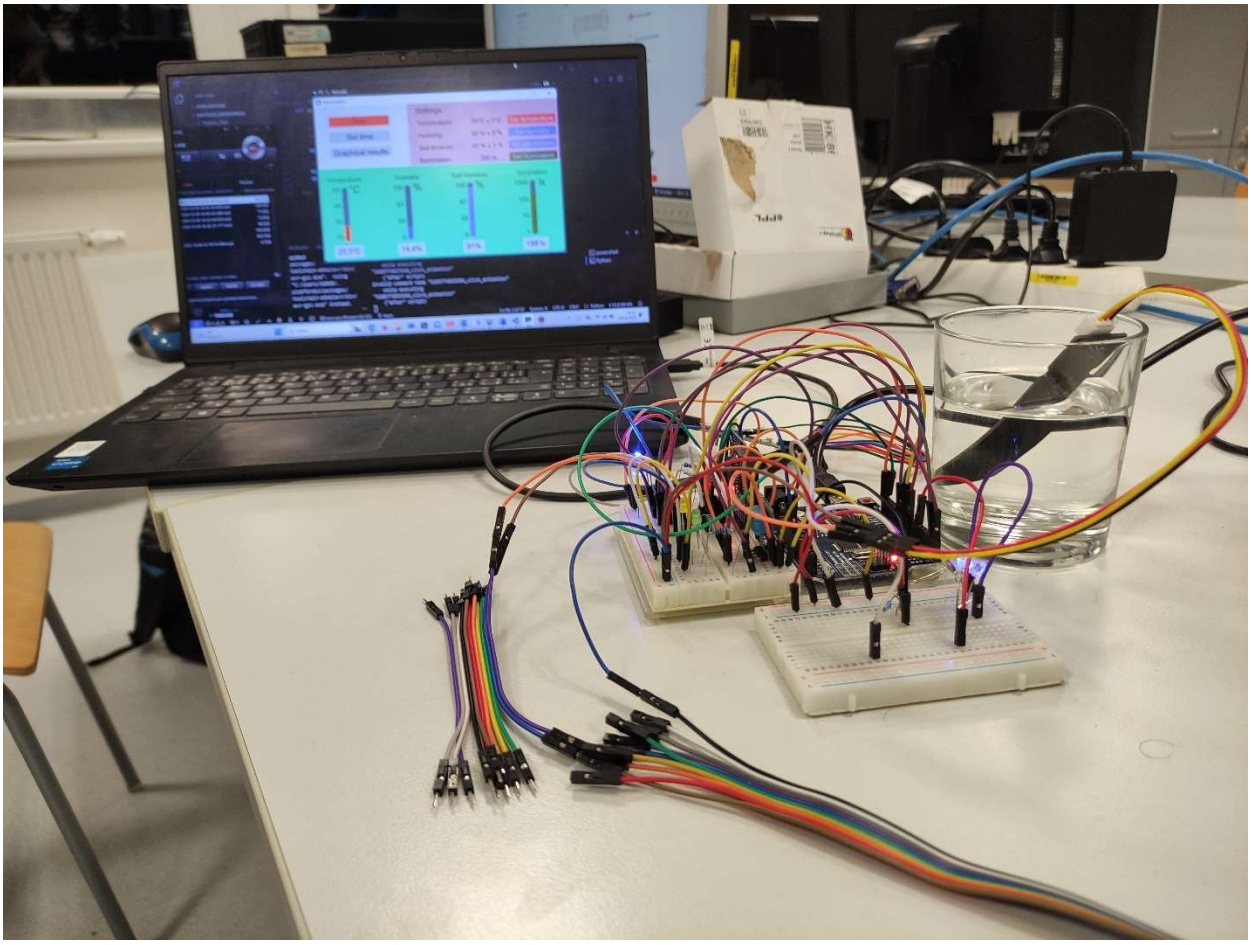
Třída GUI_main

Jedná se o třídu, která obsahuje funkce pro vytvoření GUI, obsluhu sériové komunikace a tvorbu dalších vyskakovacích oken.

Při inicializaci třídy dojde k vytvoření GUI. K tomu slouží funkce GUI_create, která volá funkce createGUImain, createGUIset a createGUIshow. Funkce createGUImain vytváří tlačítka pro připojení, zobrazení grafických výsledků a nastavení času. Funkce createGUIset generuje nabídku nastavení regulovaných veličin a createGUIshow vytvoří ukazatele naměřených hodnot.

V rozhraní se nacházejí tlačítka, při jejichž stisku se otevře nové okno:

1. *Set temperature* – okno vytvoří program temp_GUI,
2. *Set humidity* – hum_GUI,
3. *Set soil moisure* – soil_GUI,
4. *Set illumination* – ilu_GUI,
5. *Set time* – time_GUI,
6. *Graphical results* – graph_GUI.



Obr. 12 Zařízení i řídicí osobní počítač

Pro svůj chod vyžaduje hlavní okno množství funkcí:

- update_measure_data_func – zajišťuje aktualizaci hodnot zobrazených v hlavním panelu,
- update_show_settings – zajišťuje zobrazení nastavení hodnot po jejich změně,
- data_func – nastavení defaultních hodnot, tvorba datového balíčku nastavujícího regulačních veličin.

Vysílání dat z mikrokontroleru zajišťuje metoda třídy GUImain send_message(). Pokud je komunikace otevřená, dojde k předání dat sériovému portu a odvysílána.

Náročnější je přijímání dat ze sériové komunikace, protože je nutné zajišťovat chod GUI a současně kontrolovat sériový port. Funkce start_serial() otevírá sériovou komunikaci. Vytvoří vlákno, na kterém běží funkce read_from_serial(). Funkce stop_serial() komunikaci uzavírá.

Sériová komunikace

Veškerá sériová komunikace probíhá na rychlost **9600 Bd**. Je využíván sériový port počítače **COM3**. Aby se ušetřil datový tok, jsou zasílány přímo číselné hodnoty, nikoli jejich zápis v ASCII kódu. Osobní počítač vysílá dva různé datové balíčky; první nese informaci o **nastavení času**, druhý o **nastavení** regulovaných b. Mikrokontroler vysílá pouze jeden balíček.

Balíček pro nastavení času má následující strukturu:

Tab. 5 Datový balíček pro nastavení času

Byte	0	1	2	3	4	5	6	7
Obsah	'T'	Rok	Měsíc	Den v měsíci	Den v týdnu	Hodina	Minuta	Sekunda

První byte je ASCII kód písmene „T“. Slouží k rozpoznání. Ostatní byty musí formou odpovídat paměti RTC. Tento balíček je odeslán vždy, když uživatel nastaví v rozhraní čas. Balíček pro nastavení regulovaných veličin má tuto strukturu:

Tab. 6 Datový balíček pro nastavení regulovaných veličin

Byte	0	1	2	3	4
Obsah	'R'	Teplota – hodnota	Teplota – hystereze	Vlhkost – hodnota	Vlhkost – hystereze

Byte		5	6	7	8
Obsah		Půdní vlhkost – hodnota	Půdní vlhkost – hystereze	Osvětlení – stovky	Osvětlení – jednotky

První byte je ASCII kód písmene „R“. Slouží k rozpoznání. Ostatní byty obsahují přímo číselnou hodnotu, která bude nastavena na mikrokontroleru. **Hodnota** je úroveň, na kterou bude daná veličina regulována. **Hystereze** udává, s jakou tolerancí bude daná veličina nastavena. Pokud se bude veličina pohybovat v rozmezí hodnota ± hystereze, regulační členy nebudou aktivní. K odeslání dojde poté, co uživatel změní v rozhraní nastavenou regulovanou hodnotu libovolné veličiny.

Mikrokontroler každou sekundu vysílá datový balíček:

Tab. 7 Datový balíček vysílaný mikrokontrolerem

Byte	0	1	2	3	4	5	6
Obsah	Rok	Měsíc	Den v měsíci	Den v týdnu	Hodina	Minuta	Sekunda

Byte	7	8	9	10	11	12	13
Obsah	Vlhkost – celá část	Vlhkost – desetinná část	Teplota – celá část	Teplota – desetinná část	Osvětlení	Půdní vlhkost	'\n'

Veškeré časové údaje jsou ve formátu odpovídajícímu paměti RTC. Vlhkost a teplota je rozdělena na celou desetinnou část. Jedná se o data přímo získaná z teplotně-vlhkostního čidla. Dále je vysílána hodnota osvětlení a půdní vlhkosti. Je vysílána zaokrouhlená hodnota. Horních 8 bitů

z 10bitobé hodnoty odečtené z ADC. Poslední znak, nový řádek, datového balíčku označuje konec. Rozhraní vždy přečte data až do tohoto znaku. Tento znak má v ASCII tabulce hodnota 0x0A. Tato hodnota je pro ostatní byty zakázaná. Protože by teoreticky mohla nastat u hodnoty osvětlení a půdní vlhkosti, je místo této hodnoty odeslána hodnota o jedno větší. Vzniká chyba je podstatně menší než, pokud by byl datový balíček ukončen v nevhodném místě.

Zpracování přijatých dat z mikrokontroleru

Přijatá data ze sériového portu je nutné uložit, zpracovat a uložit do rozhraní, a poté jejich aplikovat aktualizaci, aby se změny projevily v hlavním okně.

Třída raw_data složí jako úložný prostor pro přijatá data. Jednotlivé byty jsou uloženy v listu. Tato třída je následně zpracována třídou data_file. Zde dochází k převodu dat na smysluplnou zobrazitelnou hodnotu. Tato data jsou ukládána do textového souboru database.txt, který slouží jako úložiště naměřených hodnot pro pozdější zpracování. Pro grafické zpracování je nutné data z textového souboru načíst. To zajišťuje třída database.

Při zpracovávání dat je nutné převádět názvy dnů v týdnu na číselné hodnoty a zpětně, stejně jako názvy měsíců na číselné hodnoty. Převod názvů dnů zajišťuje knihovna num_and_days, převod názvů měsíců na číslo funkce month2num.

Knihovna graph_func

Tato knihovna obsahuje nutné funkce pro řízení okna Graphical results. Tato knihovna obsahuje následující funkce:

- set_xlabel() – změní popis osy x grafu dle nastavené veličiny,
- set_ylabel() – změní popis osy y grafu dle nastaveného časového úseku,
- n_of_days() – vrací počet dní daného měsíce,
- n_of_days_year() – vrací počet dní daného roku,
- n_of_days_cum() – vrací počet dní, které uplynuly do začátku tohoto měsíce.

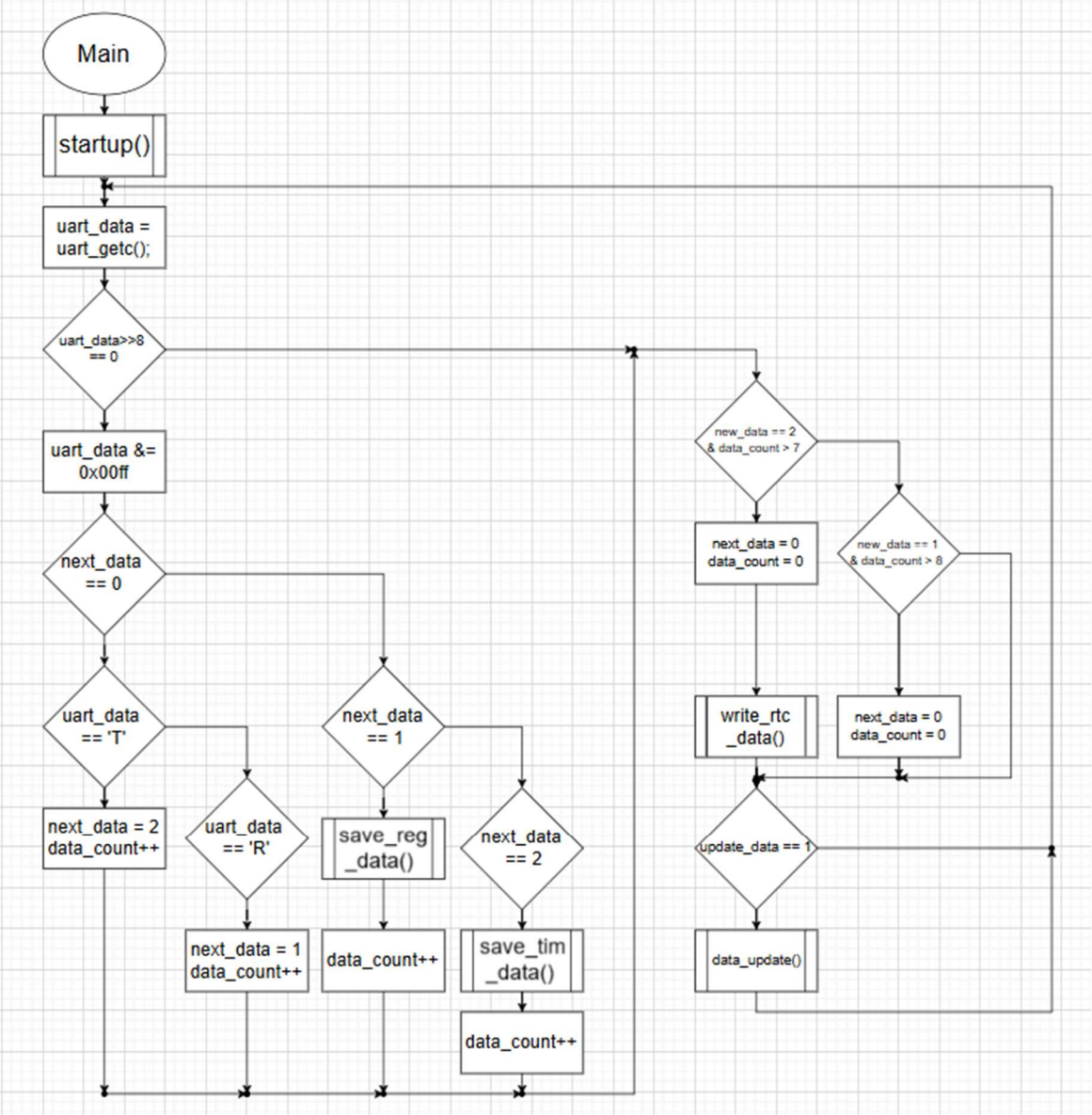
Funkce Main

Funkce musí zajistit čtení dat z teplotně-vlhkostního čidlo i hodin reálného času. Dále musí zajistit pravidelné vysílání datového balíčku a příjem dat z osobního počítače. Dále zajišťuje funkce ovládání jednotlivých regulovaných členů: topného tělesa, ventilátoru, nebulizéru, ventilu a LED pásku.

Nejdříve je volána funkce startup(), která provede potřebné inicializace periférií a nastavení defaultních hodnot globálních proměnných. Následuje nekonečná smyčka. Nejdříve jsou čtena

data ze sériové komunikace. Pokud byla nějaká data přijata, následuje zpracování přijatých dat. V opačném případě je celý tento krok přeskočen.

Funkce uart_getc() vrací 16bitovou hodnotu. Vyšší byte nenese přijatou informaci, proto je v dalším kroku odstraněn. Pokud je hodnota *next_data* nulová, jedná se o první byte datového balíčku.



Obr. 13 Vývojový diagram funkce main

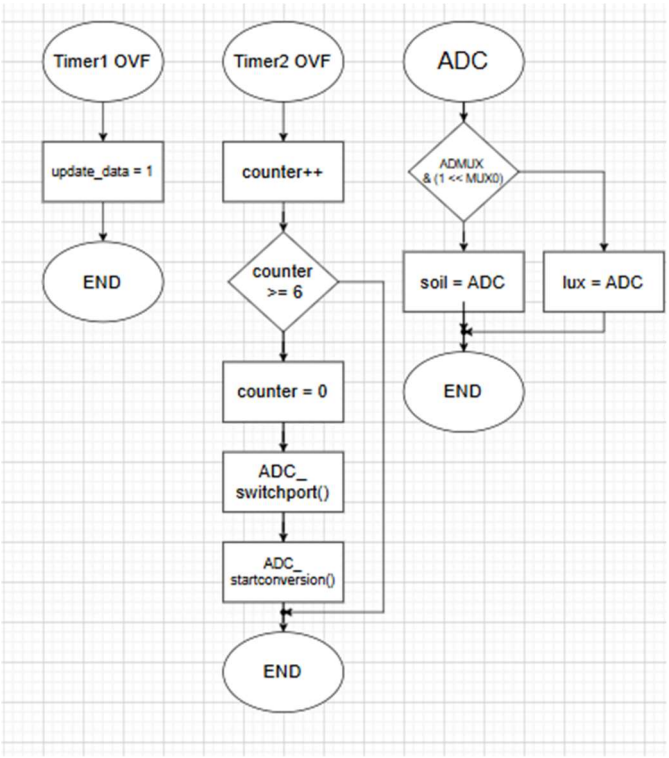
Je-li první byte ASCII kód „T“ jedná se o balíček pro nastavení času. Proměnná *next_data* je nastaven na 2 a inkrementován čítač bytů *byte_count*. Při dalším cyklu tedy program vykoná větev, pro hodnotu 2 proměnné *next_data*. Tato větev vykoná rutinu save_time_data() a následně inkrementuje čítač *byte_count*.

Je-li první byte ASCII kód „R“ jedná se o balíček pro nastavení regulace veličin. Proměnná *next_data* bude nastavena na 1. Opět je *byte_count* inkrementován. Další cyklus vykoná větev pro uložení regulovaných veličin. Bude vykonána rutina save_reg_data(). Inkrementace je shodná jako v předchozí situaci.

Následuje podmínka konce datového balíčku. Balíček pro nastavení času končí 7. bytem, balíček regulační končí 8. bytem. V obou větších jsou vynulovány proměnné *data_count* i *next_byte*. Větev pro časový balíček musí ještě proběhnout zapsání dat to hodin RTC. To zajistí funkce write_rtc_data().

Posledním krokem smyčky je kontrola, zda nedošlo k přerušení přetečením časovače Timer1. Pokud tomu tak bylo, vykoná program rutinu data_update().

Obsluha přerušení



Nezbytnou částí souboru main je obsluha přerušení. Časovač Timer1 vyvolá přerušení přibližně každou sekundu. Přerušení změní hodnotu update_data na 1, čímž v dalším cyklu spustí rutinu data_update().

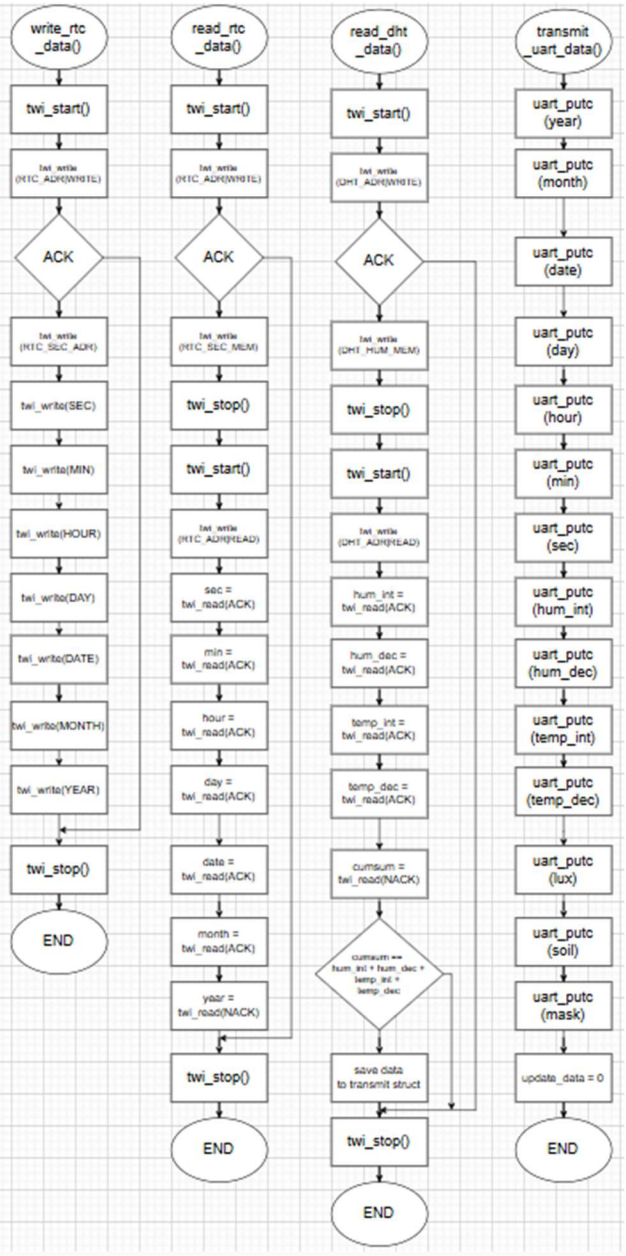
Dále je povolené přerušení AD převodníkem. Po vykování AD převodu je spuštěno. V závislosti na zvoleném analogovém vstupu je uložena buď jako půdní vlhkost, pokud probíhalo čtení z pinu PC3, nebo jako osvětlení, probíhalo-li čtení z pinu PC2.

Převod ADC zahajuje přetečení časovače Timer2. Nastane jednou za 6 přetečení časovače, který přeteče přibližně jednou za 16 ms. Mimoto dojde k přepnutí multiplexeru ADC na druhý pin.

Obr. 14 Vývojový diagram – obsluhy přerušení

Funkce pro komunikaci přes I2C

Funkce read_rtc_data() a write_rtc_data() zajištují čtení a zapisování do RTC hodin. Obě začínají startovní podmínkou a odesláním I2C adresy RTC (0x68) s požadavkem na zapisování. Další kroky jsou vykonány pouze pokud RTC komunikaci potvrdí. V opačném případě je komunikace ukončena.



Funkce write_rtc_data() pokračuje odesláním adresy paměti RTC, do které bude zapisováno; konkrétně 0x00 (sekundy). Dále jsou odesílány jednotlivé byte, které jsou v RTC uloženy. Vždy jsou přepsány buňky paměti od 0 po 6. Konkrétní podoba dat je popsán zde.

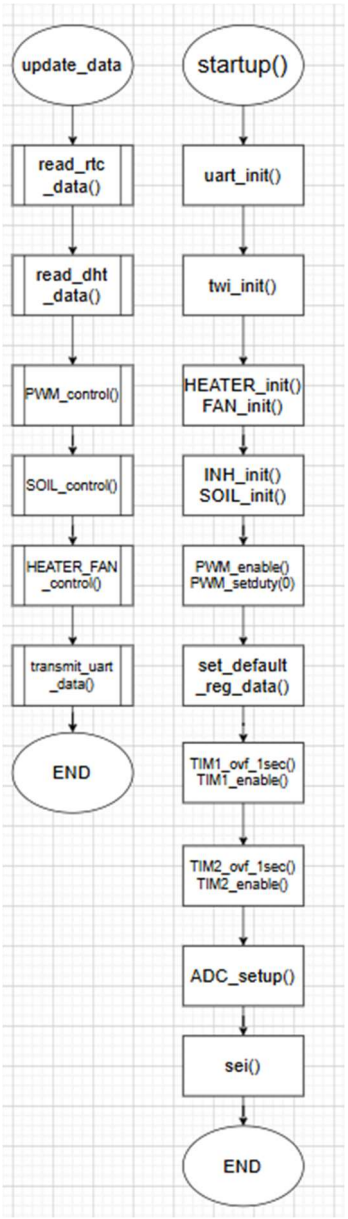
Funkce read_rtc_data() rovněž odešle adresu paměti 0x00, následuje zastavení komunikace a opakovaný start. Je odeslán další požadavek s adresou RTC, tentokráte s požadavkem na čtení. Odečtená data jsou ve shodném tvaru jako v předchozím případě. Každé čtení je potvrzeno ACK, až do adresy 0x06. Tu Master zakončí symbolem NACK, čímž je přenos ukončen. Data jsou přímo ukládána do adresáře *dp* (tzn. *data_packet*), který obsahuje data, která budou odesílána počítači.

Dále je zajištěno čtení s teplotně-vlhkostním čidlem; funkce read_dht_data(). Po startovní podmínce je odeslána adresa čidla 0x5c s požadavkem na zápis. Je-li komunikace potvrzena, obdobně jako v předchozím případě je odeslána adresa paměťové buňky 0x00 (celá část vlhkosti. Následuje opakovaný start a požadavek pro čtení. Jsou odečteny hodnoty všech pěti paměťových buněk. Formát dat je popsán zde. Data jsou uložena do dočasného úložiště. Je testována podmínka pro kumulativní součet. Je-li splněna, jsou data v adresáři *dp* přepsána.

Obr. 15 Vývojové diagramy – funkce sériové komunikace

Funkce transmit_uart_data()

Tato funkce provede odeslání datového balíčku do počítače. Obsah datového balíčku je popsán zde. Tato funkce ukončuje data update vynulováním příznaku update_data.



Funkce startup()

Tato funkce musí vykonat následující kroky. Vykoná inicializaci UARTu a I2C. Následuje inicializace jednotlivých regulovaných členů: topného členu, ventilátoru, nebulizéru i ventilu řídícího zavlažování. Při tomto kroku jsou jednotlivým periferiím též přiřazeny piny.

V dalším kroku je povoleno PWM. Střída je natavena na defaultní hodnotu 0. Zařízení ke své činnosti potřebuje mít nastavené regulované hodnoty veličin a jejich hystereze. Proto jsou v dalším kroku nastaveny hodnoty adresáře *rg* (tzn. *regulation_struct*) na defaultní hodnoty.

V dalším kroku proběhne nastavení obou časovačů a povolení přerušení při jejich přetočením. Časovač Timer1 přeteče přibližně 1 za sekundu, časovač Timer2 jednou za cca 16 ms. Posledním krokem je nastavení ADC a globální povolení všech přerušení.

Rutina data_update()

Vždy po přetečení časovače Timer1, dojde v následujícím cyklu hlavní smyčky ke spuštění rutiny data_update(). Tato rutina nejdříve odečte hodnotu z hodin reálného času; rtc_read_data(). Následuje čtení hodnot z teplotně-vlhkostního čidla; dht_rad_data(). Další funkce obsluhují regulovaná zařízení.

Funkce HEATER_FAN_CONTROL() řídí topný člen, ventilátor a nebulizér. Funkce SOIL_CONTROL() obsluhuje ventil zavlažování a funkce PWM_CONTROL() řídí intenzitu osvětlení. Nakonec proběhne odeslání dat osobnímu počítači prostřednictvím UARTu; transmit_uart_data().

Obr. 16 Vývojové diagramy – startup, data_update

Rutina save_reg_data()

Přijatá data ze sériové komunikace musí být uložena do příslušného místa v paměti mikrokontroleru, v tomto případě do adresáře *rg*. Pořadí bytů je striktní, určuje jejich význam. Je uloženo v proměnné *data_count*, a dle něj je provedeno uložení bytu.

Tab. 8 Uložení bytů balíčku nastavení regulovaných veličin

Data_count	Úložiště
1	*temp_val*
2	*temp_hys*
3	*hum_val*
4	*hum_hys*
5	*soil_val*
6	*soil_hys*
7	*ilu_100*
8	*ilu_1*

Rutina save_time_data()

Shodným způsobem jsou uložena data časového balíčku. V tomto případě jsou data uložena v adresáři *rc* (tzn. *real-time clock struct*).

Tab. 9 Uložení bytů balíčku nastavení času

Data_count	Úložiště
1	*year*
2	*month*
3	*date*
4	*day*
5	*hour*
6	*minute*
7	*second*

Knihovny

Firmware mikrokontroleru využívá několik knihoven. Knihovna uart byla převzata z [19]. Použitá verze byla převzata z [20]. Knihovna twi byla převzata z [21]. Ani jedna z nich nebyla upravována. Ostatní knihovny byly vytvořeny spolu projektem. Podrobné informace k jednotlivým knihovnám jsou obsaženy v dokumentaci.

Heater

Knihovna heater slouží k automatizovanému řízení topení, ventilátoru a nebulizéru na základě aktuálních hodnot teploty a vlhkosti a nastavených pruhů zapínaní a vypínání jednotlivých zařízení. Nastavení prahů je provedena pomocí hystereze, která zabraňuje častému zapínání a vypínání zařízení. V opačném případě by jinak mohlo docházet k neefektivnímu chodu. Stav jednotlivých zařízení je uloženo v proměnné sw_flat, která používá jednotlivé bity k inicializaci, která zařízení jsou aktuálně zapnuta. K zajištění řízení jednotlivých periferií slouží funkce HEATER_FAN_control,

Funkce HEATER_FAN_control

Funkce zajišťuje řízení těchto periférií: topného členu, ventilátoru a nebulizéru.

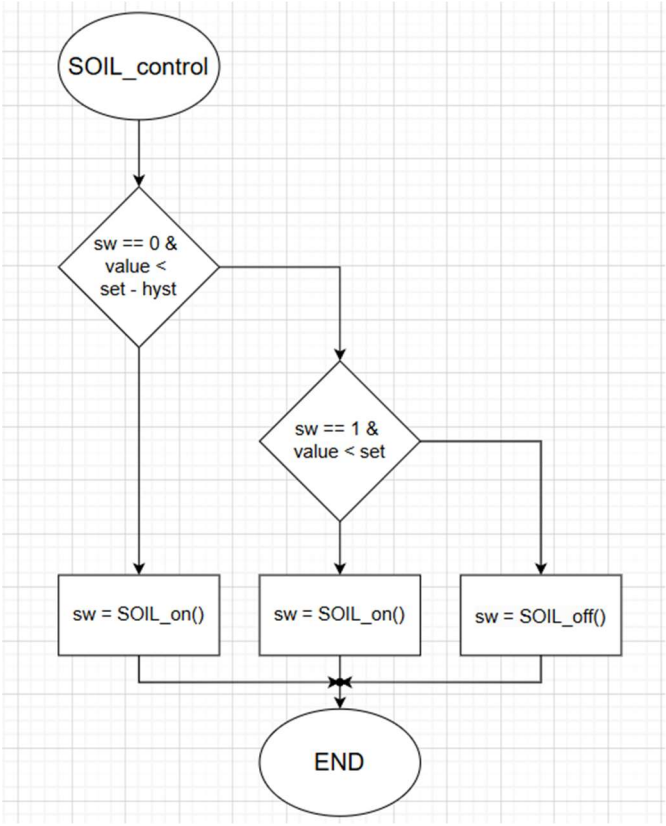
Tab. 10 Tabulka spuštěných periférií

		Vlhkost		
		< Set	≈ Set	> Set
Teplota	< Set	T+N	T	T
	≈ Set	N	X	V
	> Set	V	V	V

V tabulce 10: **T** značí **topný člen**, **V** značí **ventilátor**, **N** značí **nebulizér** a **X** stav **vypnuto**. Obecně má prioritu teplota před vlhkostí. Zakázaný je souběh ventilátoru a nebulizéru, dále ventilátoru a topného členu. Zařízení se spouští, klesne-li (či stoupne-li) hodnota pod nastavenou hodnotu zmenšenou (zvětšenou) o hysterezi. Zapnuto zůstane až do dosažení jmenovité hodnoty.

Soil

Tato knihovna slouží k automatickému zavlažování. To se provádí řízením ventilu za pomoci hystereze na základě měření vlhkosti půdy. Ventil se zapne, pokud bude hodnota menší než nastavená hodnota mínus hystereze. Ventil zůstává otevřen i v moment, kdy je hodnota menší než nastavená hodnota. V opačném případě se ventil vypne, pokud bude hodnota vyšší.



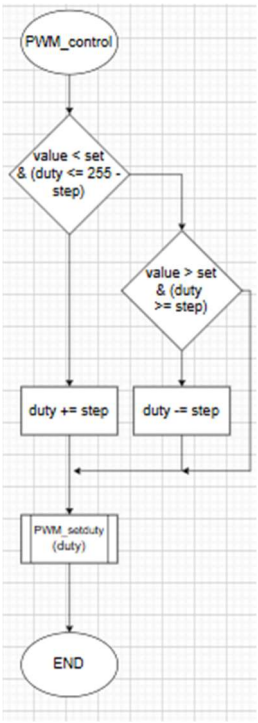
Obr. 17 Vývojový diagram SOIL_control

PWM

Knihovna **PWM** obsahuje funkce pro konfiguraci a řízení PWM:

- PWM_enable() – aktivuje PWM na časovači 0, nastaví režim Fast PWM a neinvertovaný výstup na pinu OC0A (PD6). Časovač je spuštěn s předělovačem (prescalerem) 64.
- PWM_set_duty(duty) – nastavuje pracovní cyklus PWM. Parametr duty (typu volatile uint8_t) je hodnota v rozsahu 0–255, kde 0 odpovídá 0 % a 255 odpovídá 100 %. Hodnota je nastavena do registru OCR0A.
- PWM_control(current_value, set_value, step) – dynamicky upravuje pracovní cyklus PWM na základě aktuální hodnoty current_value (typu uint16_t), požadované hodnoty set_value (typu uint16_t) a kroku změny step (typu uint8_t). Funkce zvyšuje nebo snižuje pracovní cyklus podle rozdílu mezi aktuální a požadovanou hodnotou, přičemž chrání hodnotu před přetečením nebo podtečením.

Funkce **PWM_enable()** inicializuje PWM a musí být volána před nastavením pracovního cyklu. Funkce **PWM_set_duty()** umožňuje přímé nastavení pracovního cyklu například pro pevné řízení intenzity LED nebo rychlosti motoru. Funkce **PWM_control()** je použita pro adaptivní řízení pracovní cyklu na základě aktuální a požadované hodnoty, což umožňuje plynulou regulaci intenzity výstupu. Všechny funkce společně umožňují efektivní řízení PWM v různých aplikacích.



Obr. 18 Vývojový diagram – PWM_control

ADC

Knihovna ADC obsahuje funkce pro nastavení čtení dat:

- ADC_setup() – nastaví potřebné bity pro funkci ADC
- ADC_startconversion() – spustí ADC konverzi
- ADC_switchport() – změní čtení hodnoty mezi piny 4 a 5

Funkce ADC_setup() nastaví tři registry na požadované hodnoty pro povolení ADC. Registr ADCSRA nám nastaví předděličku na 32 (ADPS2 a ADPS1 bity) pro kontrolu hodinové rychlosti ADC. Registr ADMUX nastaví referenční hodnotu na Vcc (bit REFS0) a čtení z pinu 5 (bity MUX0…3). Registr ADCSRA povolí ADC (bit ADEN) a ADC interrupt (bit ADIE).

Funkce ADC_startconversion() spustí ADC konverzi nastavením ADSC bitu v registru ADCSRA. Toto spustí konverzi na vybraném analogovém vstupu.

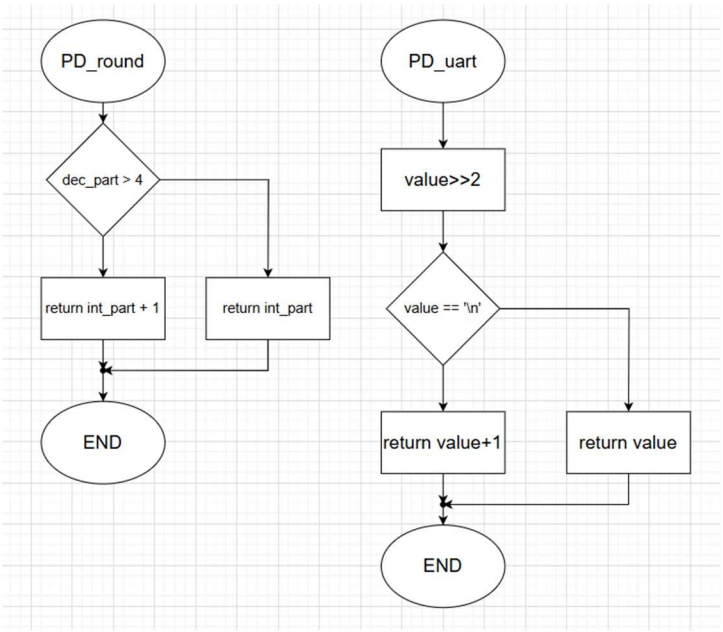
Funkce ADC_switchport() přepíná mezi piny 4 a 5 změnou bitu MUX0 v ADMUX registru.

Preper_data

Knihovna Preper_data obsahuje funkce provádějící opakující se výpočty:

- PD_round()– zaokrouhluje změřenou hodnotu teploty nebo vlhkosti na celé číslo,
- PD_uart() – 10bitovou hodnotu osvětlení a půdní vlhkosti převede na 8bitovou,
- PD_ilu() – obsahuje přepočet hodnoty z ADC pro osvětlení (viz Hardwarová popis),
- PD_soil() – obsahuje přepočet hodnoty z ADC pro půdní vlhkost (viz Hardwarová popis).

Funkce PD_round() je aplikována na vstupní parametry *current_temp* a *current_hum* funkce HEATER_FAN_control(). Funkce PD_uart() upraví odesílanou hodnotu přes UART. Funkce PD_ilu() je použita pro úpravu vstupní proměnné *current_value* funkce PWM_control() a funkce PD_soil() pro vstupní proměnou *current_soil* funkce SOIL_control(). V obou případech jsou požity, aby nastavená a změřená hodnota měly stejný formát a bylo je možné porovnávat.



Obr. 19 Vývojové diagramy – funkce PD_round a PD_uart

Instrukční list

Oživení zařízení

1. Sestavené zařízení nebude plnit svou funkci, nebude-li do mikrokontroleru nahrán program Project-02-03. Při nahrávání programu musí být převodník CP2102 odpojen.
2. Připojte převodník.
3. Spusťte aplikaci Tropical plants.
4. Stisknutím tlačítka *Connect* dojde ke spojení počítače se zařízením. Tlačítko *Connect* se změní na *Run*. Stiskněte pro zahájení komunikace. Další stisk ukončí spojení.
5. Nastavte čas na zařízení stisknutím tlačítka *Set time*. Je nutné vyplnit všechny políčka. Nastavený čas potvrdíte stiskem tlačítka *Apply*. Tlačítkem *Close* se vrátíte zpět.

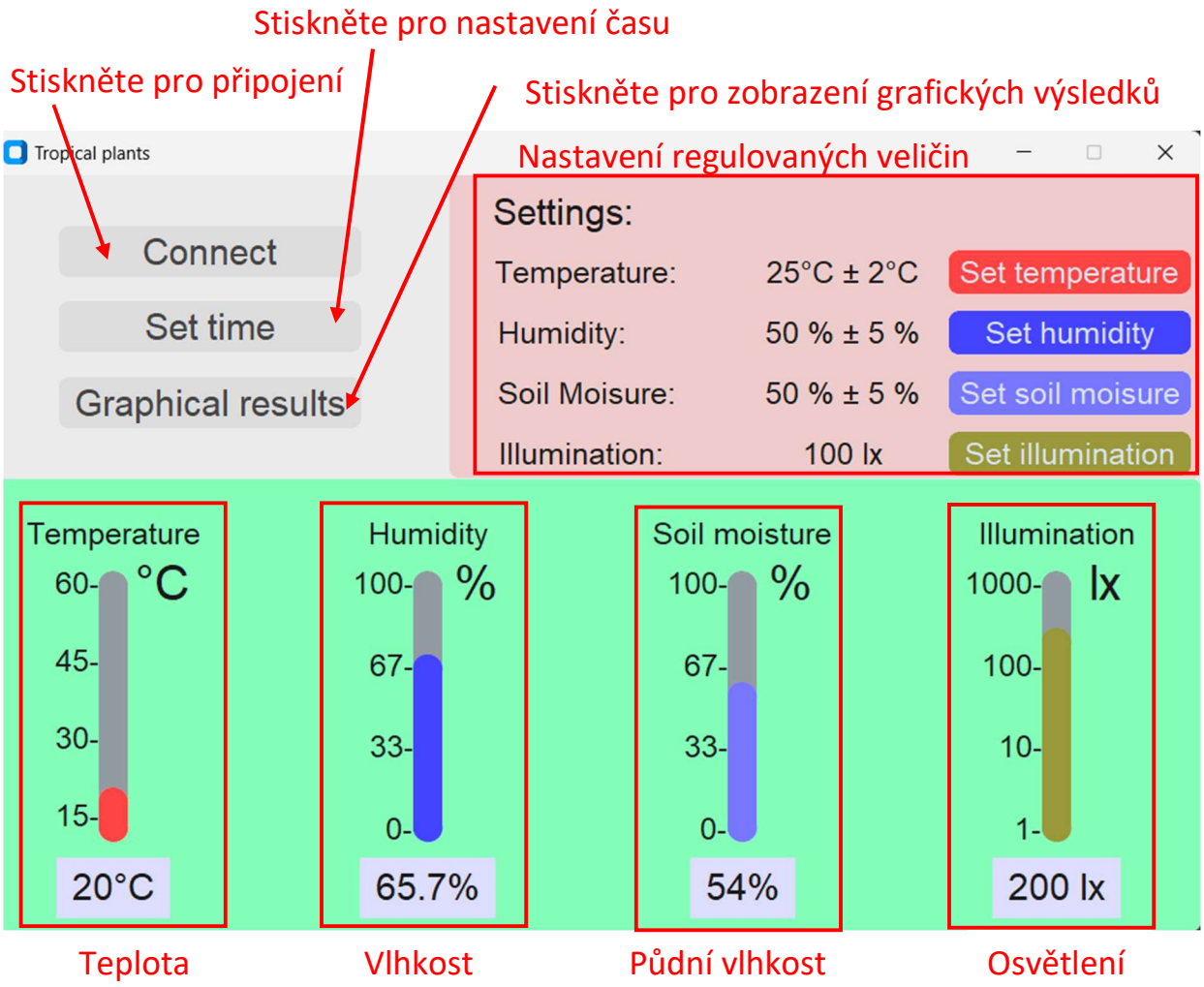
Zobrazení aktuálně naměřené hodnoty

Naměřené hodnoty jsou přímo zobrazeny v hlavním panelu. Hlavní panel zobrazuje naměřenou hodnotu teploty, vlhkosti vzduch, půdní vlhkosti a osvětlenosti. U všech veličin je jednak zobrazena číselná hodnota a jednak její hodnota na analogové stupnici.

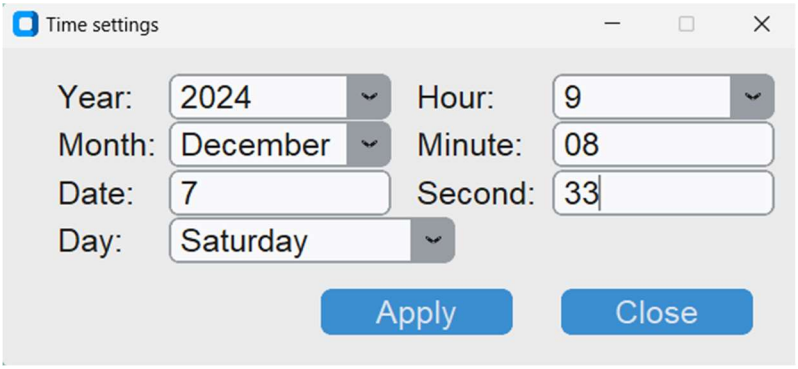
Analogové stupnice mají následující rozsah:

- Teplota – 15 až 60 °C
- Vlhkost vzduchu – 0 až 100 %
- Půdní vlhkost – 0 až 100 %
- Osvětlení – 1 až 1000 lx

Stupnice osvětlení je logaritmická, ostatní jsou lineární.

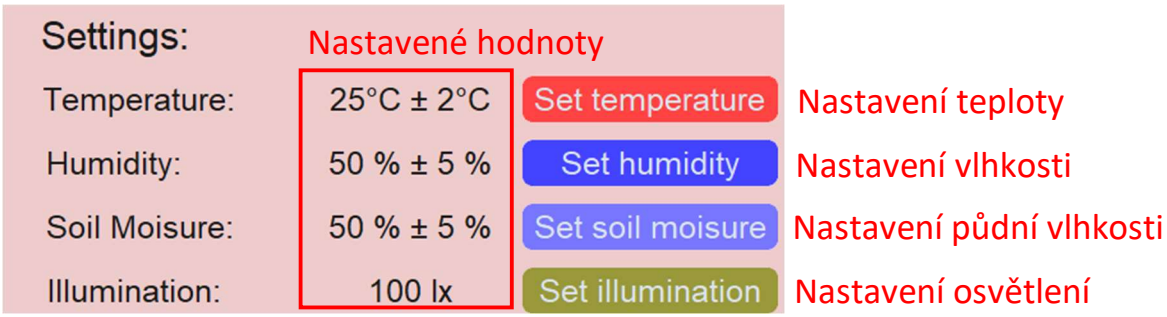


Obr. 20 Hlavní panel aplikace Tropical plants



Obr. 21 Panel nastavení času

Nastavení regulovaných veličin



Obr. 22 Panel nastavení

Po spuštění zařízení nastaveny tyto hodnoty veličin:

- Teplota – 25°C, hystereze 2°C,
- Vlhkost vzduchu – 50 %, hystereze 5 %,
- Půdní vlhkost – 50 %, hystereze 5 %,
- Osvětlenost – 100 lx.

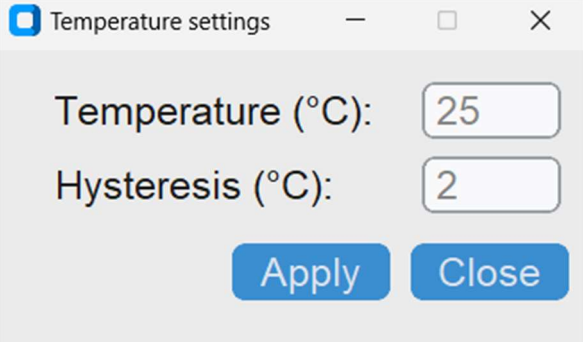
Hystereze stanovuje, v jakém rozmezí bude hodnota kolísat.

Nastavení veličin lze měnit následujícím způsobem:

- Teplota – Tlačítko *Set temperature*
  - Hodnota: 15 až 50°C
  - Hystereze: 1 až 10°C
- Vlhkost vzduchu – Tlačítko *Set humidity*
  - Hodnota: 20 až 90 %
  - Hystereze: 5 až 30 %
- Půdní vlhkost – Tlačítko *Set soil moisure*
  - Hodnota: 5 až 90 %
  - Hystereze: 5 až 30 %
- Osvětlenost – Tlačítko *Set illumination*
  - Hodnota: 20 až 500 lx

Hodnotu napište do příslušného pole a potvrďte stiskem *Apply*. Okno uzavřete stiskem *Close*. Pokud byla změna provedena dojde k přepsání nastavených hodnot v hlavním panelu. Změna nenastane, pokud ani jedna z nastavených hodnot nebude v povoleném rozsahu. Pokud alespoň jedna z hodnot bude platná, přepíše se pouze ta, která je platná. Změny se okamžitě projeví na zařízení.

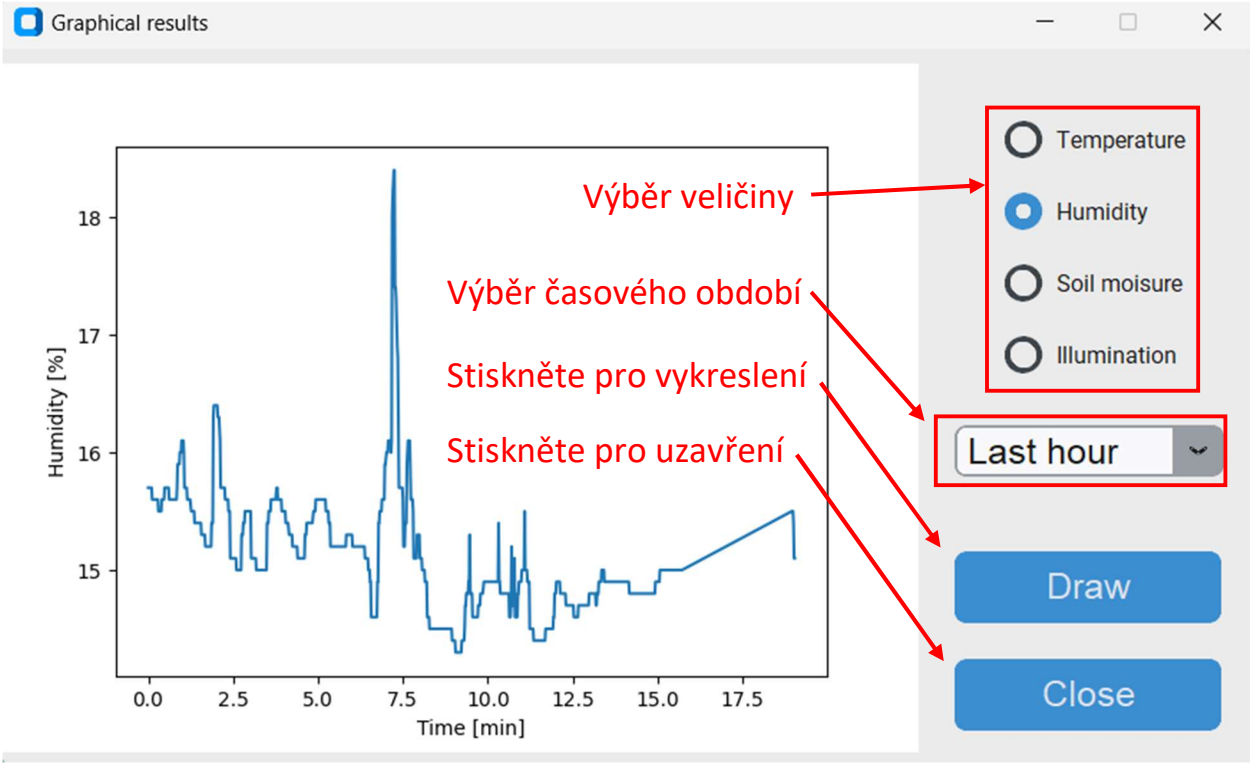
Povolené meze nastavení je možné změnit v programu data_func, ve funkci default_data(), v adresáři systém_settings. Meze měňte obezřetně. V žádném případě nesmí nastat situace, že by veličiny byly nastavené s nulovou hysterezí.



Obr. 23 Panel pro nastavení teploty

Grafické zobrazení naměřených hodnot

Program Tropical plants umožňuje též zobrazit vývoj veličin za poslední časové období.



Obr. 24 Panel pro zobrazení grafických výsledků

Zvolte veličinu, která bude zobrazena:

- *Temperature* – teplota,
- *Humidity* – vlhkost vzduchu,
- *Soil moisure* – půdní vlhkost,
- *Illumination* – osvětlenost.

Zvolte časové období:

- *Last minute* – poslední minuta,
- *Last hour* – poslední hodina,
- *Last day* – poslední den,
- *Last week* – poslední týden,
- *Last month* – poslední měsíc
- *Last year* – poslední rok,
- *All* – veškerá dostupná data.

Stiskněte *Draw* pro vykreslení grafu. Stiskem *Close* pro uzavření okna.

Problémy s připojením

Pokud není možné zařízení připojit tlačítko Connect se podbarví červeně. Za těchto okolnostní učiňte tyto kroky:

1. Zkontrolujte, zda je zařízení připojeno.
2. Vyjměte microUSB konektor z převodníku a restartujte aplikaci Tropical plants.
3. Zkontrolujte, zda Váš osobní počítač využívá sériový port COM3. Není-li tomu tak, nastavte používaný port ve zdrojovém kódu Tropical_plants.
4. Zkontrolujte, zda osobní počítač umožňuje komunikaci přes sériový port. Není-li tomu tak nainstalujte příslušný ovladač [18].

Ukázka obsluhy

Pro ukázku obsluhy zařízení a jeho činnosti za chodu klikněte zde.

Reference

1. Climate Chamber System.
2. Learning AVR-C Episode 7: PWM.
3. Learning AVR-C Episode 8: Analog Input.
4. Custom Tkinter - Official Documentation
5. pySerial's documentation
6. ASCII table
7. ATMEGA328P- datasheet
8. Soil moisure
9. Arduino map()
10. Co potřebují rostliny k životu - Jaké jsou podmínky pro jejich život | Zjišťujeme.cz
11. Podnebné (klimatické) pásy - Počasí
12. Your Gateway to Embedded Software Development Excellence · PlatformIO
13. DS3213 datasheet
14. Soil Moisture Sensor - Comple Guide | Arduino Project Hub
15. DHT12 temperature sensor and Arduino example - Arduino Learning
16. Arduino - Home
17. CP2102 datasheet(1/18 Pages) SILABS | SINGLE-CHIP USB TO UART BRIDGE
18. CP210x USB to UART Bridge VCP Drivers - Silicon Labs
19. Peter Fleury Online: AVR Software
20. AVR course – UART · tomas-fryza/avr-course
21. AVR course – I2C · tomas-fryza/avr-course

Seznam použitých nástrojů

1. VS Code
2. ChatGPT
3. Microsoft Copilot
4. SimulIDE
5. Saleae Logic 2
6. Online C compiler
7. Bandicam
8. draw.io
9. Doxygen
10. Matlab
11. ProfiCAD - Elektro CAD Software - ProfiCAD