Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Automatická zahrádka ovládána platformou Arduino

Jindřich Cvak

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jindřich Cvak

Informatika

Název práce

Automatická zahrádka ovládána platformou Arduino

Název anglicky

Automatic garden controlled by Arduino

Cíle práce

Cílem teoretické části je analýza dostupných komponentů včetně porovnání i jiných platforem mimo arduino. Zahrnuto bude i porovnání s jinými podobnými projekty.
Po zpracování této části by měly být vybrány ideální dostupné komponenty pro teoretický a později i praktický návrh projektu.
Zapotřebí je také analyzovat správu rostlin, aby byl projekt po sestavení a nastavení schopen je obhospodařit.
Dále je potřeba navrhnout vhodné zapojení a udělat teoretický návrh programu.

Cílem praktické části práce je sestrojit a naprogramovat vzorový, automatický, nastavitelný systém na správu rostlin.

Metodika

Prvním krokem bude studium samotné platformy arduino. Od prostředí až po jednotlivé příkazy.

Druhým krokem bude analýza a výběr komponentů pro samotný projekt včetně studia specifikací.

Třetím krokem bude návrh a sestavení samotného projektu.

Čtvrtým krokem bude návrh a implementace programu.

Pátým a posledním krokem bude nastavení celého projektu pro potřeby určité rostliny.

Doporučený rozsah práce

30-60 stran

Klíčová slova

arduino, zahrada, automatický systém, správa rostlin, program

---

Doporučené zdroje informací

POKLUDA, R. – KOBZA, F. *Skleníky, fóliovníky, využití a pěstební technologie*. Praha: Profi press, 2011. ISBN 978-80-86726-46-5.

Voda, Z – tým HW Kitchen. Průvodce světem Arduina 2. vydání. Bučovice: Nakladatelství Martin Stříž, 2017. ISBN: 978-80-87106-93-8

Předběžný termín obhajoby

2021/22 LS – PEF

Vedoucí práce

Ing. Dana Vynikarová, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 1. 3. 2022

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 7. 3. 2022

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 14. 03. 2022

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci " Automatická zahrádka ovládána platformou 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 15.3.2022 _____

Poděkování

Rád bych touto cestou poděkoval své neuvěřitelně trpělivé a ochotné vedoucí Ing. Daně Vynikarové Ph.D. za podporu a vedení při práci.

Automatická zahrádka ovládána platformou Arduino

Abstrakt

Bakalářská práce se zabývá problematikou skleníků a jejich automatizace za účelem snížení rizika lidského faktoru na růst rostlin. V prvních částech práce jsou objasněny důvody automatizace a užívaní skleníků. Tyto poznatky budou dále použity při praktické časti. Dále jsou teoreticky popsány jednotlivé komponenty, které budou v projektu použity.

Praktická část popisuje mechanickou ale i programovou část projektu, která je podrobně rozepsána. Výsledkem praktické části je sestavený model systému, který bude schopen postarat se o rostliny. Při testování byly odhaleny různé problémy. Například při čtení dat z displeje, kdy se do programu propisovaly špatné hodnoty, které se ale podařilo do značné míry programově potlačit, anebo koroze odporového čidla vlhkosti půdy, které bylo vyměněno za kapacitní.

Klíčová slova: Arduino, chytrý skleník, automatizace skleníku, mikrokontroler ATmega2560, sekvenční programováni, jazyk Wiring

Automatic garden controlled by Arduino

Abstract

The bachelor thesis deals with the issue of greenhouses and their automation in order to reduce the risk of the human factor on plant growth. The first parts of the thesis explain the reasons for automation and the use of greenhouses. This knowledge will be further used in the practical part. Furthermore, the individual components that will be used in the project are theoretically described.

The practical part describes the mechanical as well as the program part of the project, which is described in detail. The result of the practical part is a compiled model of the system that will be able to take care of the plants. Various problems were identified during testing. For example, when reading data from the display, when bad values were written into the program, which was largely suppressed programmatically, or corrosion of a resistance soil moisture sensor, which was replaced by capacitive.

Keywords: Arduino, smart greenhouse, greenhouse automation, microcontroller ATmega2560, sequential programming, Wiring language

Obsah

Seznam obrázků

1 Úvod

Lidé postupem času ztrácejí zájem o manuální činnosti, například velkou část zemědělství v dnešní době zastupují stroje, ať jsou to sklízecí mlátičky (kombajny), sušičky nebo jiné stroje, které jsou ve svém odvětví mnohem rychlejší a přesnější než lidé. Lepší podmínky se dají poskytnout i rostlinám, co teprve rostou, aby rostly rychleji, měly lepší a větší plody, nebo aby se mohli sázet i v oblastech, kde nemají své přirozené podmínky.

Rozhodl jsem se sestavit a naprogramovat automatický systém, který by se dokázal o rostliny sám postarat vykonáváním činností jako jsou zalévání, osvětlení a větrání.

K tomuto tématu mě přivedl můj děda, který od jara do podzimu tráví většinu času na chalupě, kde pěstuje různorodé ovoce a zeleninu. Nespočetněkrát se stalo, že mu úrodu poničilo nepříznivé počasí, které způsobilo změnu podmínek u rostlin a jejich následné zplesnivění, spálení či uschnutí.

Napadlo mě, že všem těmto problémům by se dalo předejít včasným zakročením. Nedává rozum, aby kdokoliv hlídal rostliny 24 hodin 7 dní v týdnu, když je může sám a automaticky hlídat správně zapojený, naprogramovaný a nastavený mikrokontroler se správnými periferiemi Jelikož mám k tomuto tématu blízko, rozhodl jsem se pro realizaci mého nápadu. Má práce by mohla inspirovat jiné lidi a vzbudit v nich zájem a programování mikrokontrolerů. Možnosti jsou zde velice široké, záleží jen na dostupných periferiích a zkušenostech.

Čtenář bude v práci seznámen s automatizací a jejími dopady. Se skleníky, jejich vlastnostmi a tím, jaký mají účinek na rostliny. S Aruinem a platformami jemu podobnými. S určitými Arduino deskami a vývojovým prostředím, kde se programují. Dále bude čtenář obeznámen s jednotlivými komponenty, které budou v projektu použity. S jejich celkovým zapojením do funkčního celku a následným podrobným popisem naprogramování a nastavení.

2 Cíl práce a metodika

2.1 Cíle práce

Cílem teoretické části je analýza dostupných komponentů včetně porovnání i jiných platforem mimo Arduino. Zahrnuto bude i porovnání s jinými podobnými projekty. Po zpracování této části by měly být vybrány ideální dostupné komponenty pro teoretický a později i praktický návrh projektu.

Zapotřebí je také analyzovat správu rostlin, aby byl projekt po sestavení a nastavení schopen je obhospodařit.

Dále je potřeba navrhnout vhodné zapojení a udělat teoretický návrh programu.

Cílem praktické části práce je sestrojit a naprogramovat vzorový, automaticky nastavitelný systém na správu rostlin.

2.2 Metodika

Prvním krokem bude studium samotné platformy Arduino. Od prostředí až po jednotlivé příkazy.

Druhým krokem bude analýza a výběr komponentů pro samotný projekt včetně studia specifikací.

Třetím krokem bude návrh a sestavení samotného projektu.

Čtvrtým krokem bude návrh a implementace programu.

Pátým a posledním krokem bude nastavení celého projektu pro potřeby určité rostliny.

3 Automatizace

Tato kapitola se týká teoretické části, kdy se čtenář seznámí s automatizací, její historií a vysvětlením, proč se automatizuje a jaké to přináší důsledky.

„Automatizace ve zkratce je řízení bez zásahu člověka, obecně se tak myslí minimální účast člověka na řízení, označuje se tak použití samořídících systémů k řízení technologických zařízení a procesů“ [1]

3.1 Historie automatizace

Počátky automatizace můžeme vidět už kolem roku 200 př. n. l. v Alexadrii, kde přišli na způsob, jak pomocí páry otevírat a zavírat velké a těžké chrámové dveře. Šlo o způsob využití protizávaží, což bylo ve své době skutečně velkým vynálezem. [2]

Dalším pokrokovým strojem byl takzvaný samotřas, který sloužil k přísunu zrna ve starých mlýnech a kolem 9. století se objevily stroje, které ovládaly přítok vody do napáječek pro dobytek. [2]

Ke skutečné automatizaci došlo až v roce 1801, kdy Francouz Ch. Jackquard sestrojil dopřádací stroj, který se začal používat v průmyslu. V následujících desetiletí se začaly objevovat revolverové soustruhy, papírenské stroje a zemědělské mlátičky.

V roce 1914 vznikla ve společnosti Ford první montážní linka, která zkrátila tehdejší proces výroby z 14 h na 6 h. Stále ale bylo zapotřebí, aby u práce byli lidé, kteří se starali o upínání, vyjímání a přenášení výrobků. [2]

Před druhou světovou válkou začaly vznikat stroje, které již skoro nepotřebovaly lidský zásah a byly schopné pracovat téměř samostatně. V těchto letech se začaly publikovat první teoretické práce popisující principy automatického řízení. [2]

V roce 1954 přišel na svět první plně automatický NC systém, který byl předlouhou pro dnešní průmyslové roboty, CNC stroje a automatizaci takovou, jakou ji známe dnes. V dnešní době jsou tyto stroje naprosto nepostradatelné a jejich uplatnění nalezneme takřka v každém odvětví. [2]

3.2 Důvody a důsledky automatizace

S automatizací se v dnešní době setkáváme prakticky neustále. Zjednodušuje nám každodenní rutinu a zbavuje nás povinnosti vykonávat jednoduché i složitější repetitivní činnosti. S dostatkem času, nápadů a financí lze automatizovat téměř jakoukoliv činnost,

která nevyžaduje lidský faktor (tvořivost, představivost, emoce). Od samotřasu, přes chytré zahrádky, až po velké automatizované výrobní linky nebo velké 3D tiskárny co dokáží tisknot domy. S tímto ulehčením práce pak lidem zbývá více času na řešení jiných, třeba i komplexnějších a smysluplnějších problémů. V jiných případech je toto ulehčení jen zpříjemněním a zjednodušením každodenního života. Za hlavní výhody automatizace je brána úspora času, peněz, lidských zdrojů, vymanění se lidské chybovosti a všeobecně lepší přehled a kontrola nad prací. [3]

Toto usnadňování má ale i své nepříliš světlé stránky. Nadměrná automatizace způsobuje, že na ní lidé začínají být závislí. Vezměme si za příklad prosté vytloukání obilného zrna, to v dnešní době obyčejní lidí neumí. To by mohlo mít kritické následky v případě celosvětových katastrof, války, blackoutu atp. Lidé kvůli automatizaci přicházejí o schopnost naprosté soběstačnosti a o schopnost přežití bez technologií.

4 Skleníky

Tato kapitola teoretické části pojednává o sklenících a jejich vlastnostech v návaznosti na potřeby rostlin.

Skleník je stavba se skleněnými stěnami, popřípadě foliovými, která má za úkol udržovat v prostoru uvnitř stálou teplotu. Důvodů proč se skleníky budují je spousta. Například se v nich dají pěstovat rostliny, které by se za normálních podmínek v dané oblasti pěstovat nedali. Zlepšují podmínky pro lepší plodiny. Používají se při šlechtění nových rostlin a také v jarních a podzimních měsících chrání rostliny před nečekanými mrazy. [4]

Automatizovaný skleník má oproti obyčejnému spousty výhod. Jako je například automatické zalévání, přisvicování v případech, kdy je zataženo anebo regulaci teploty uvnitř pomocí odvětrávání, případného topení. Díky tomu takové skleníky dokáží zajistit ideální podmínky pro pěstovaní rostlin. [5]

4.1 Skleníkový efekt

Skleníkový efekt slýcháme nejčastěji ve spojení s globálním oteplováním. U skleníků to funguje velice podobně. Světlo dopadá skrz stěny do skleníku, kde se mění z části na teplo. Jelikož je ale skleník uzavřeným prostorem, teplo se v něm kumuluje. To má za následek udržování nebo vzrůst teploty uvnitř. [5]

4.2 Historie skleníků

Vznik skleníků se připisuje starému Římu. Kde císař Tiberius natolik miloval dováženou zeleninu s názvem arménská okurka, která se v klimatických podmínkách Říma nedala pěstovat, že zřídil takzvané specularium. Stavba potáhlá bílou plachtou a impregnována olejem fungovala jako improvizovaný skleník.

Skleníky, tak jak je známe dnes, vzniky v Itálii kolem 13 století a sloužili jako botanické zahrady, pro pěstování exotických rostlin. Odtud se začali postupně šířit do celého světa. [6]

4.3 Potřeby rostlin

Je známo že rostliny ke svému růstu a prosperitě požadují dostatek světla, tepla, dostatečně vlhkou půdu, dostatečně vlhký okolní vzduch. Avšak nadměrné množství těchto

vlivů může rostlinám uškodit nebo je dokonce zahubit. Každé rostlině vyhovují trošku jiné podmínky okolí. [7]

4.3.1 Vlhkost půdy

Vlhkost půdy je jedním z nejdůležitějších faktorů. Například pro bylinky je ideální vlhkost půdy kolem 50 %. Zatímco pro zeleninu je 50 % krajních. Klesne-li hodnota vlhkosti půdy u zeleniny pod 50 % může to mít pro rostliny velice nepříznivé následky. [7]

4.3.2 Teplota

Co se týče teploty, jak u bylinek, tak zeleniny by měla být kolem 20 °C. Teplota má zásadní dopad na fotosyntézu. Při vyšších teplotách je fotosyntéza rychlejší, a to až několikanásobně. [7]

4.3.3 Světlo

Světlo je pro rostliny velice důležitým faktorem a má přímou spojitost s jejich růstem a vývojem. Doporučený počet hodin světla za den se pohybuje kolem 10–12 hodin. [7]

5 Arduino

Tato kapitola se týká teoretické části, kdy je čtenář seznámen s platformou Arduino, s jeho historií, s dostupnými deskami, včetně porovnání s jinými platformami a odůvodněním proč si vybrat právě Arduino.

Arduino je známo díky své jednoduchosti a přívětivosti pro nové uživatele. To má za následek velkou rozšířenost této platformy. [8]

Arduno používá procesory od firmy Atmel a k desce je možné zapojit velké množství vstupních a výstupních periferii jako jsou například čidla, čerpadla, serva atp. [8]

Programovací jazyk, ve kterém jsou psány programy pro Arduino se nazývá Wiring a vychází z jazyka C++. Arduino se take vyznačuje velkou rozšířeností a podporou na fórech, kde lze nalézt nepřeberné množství informací a rad, jak řešit určité problémy. Mezi nejznámější desky platformy Arduino patří Arduino Uno Rev3, Arduino Mega2560 Rev3 a Arduino Leonardo. [8]

5.1 Historie platformy Arduino

„Projekt Arduino vznikl v roce 2005 v Itálii, přímo na deskách si můžete všimnout nápisu Made in Italy. Původní účel však směřoval na studenty. Mělo jít o jednoduchou prototyp platformu, která umožní rychlý vývoj a velmi jednoduché používání. K jeho oblíbenosti se začaly vydávat i novější verze, a tak se v roce 2010 prodalo více než 120 000 kusů. V tomto roce také vyšla jedna z přelomových desek. Deska měla označení UNO a je, dá se říci jednou z nejpoužívanějších.“ [9]

5.2 Porovnáni s jinými platformami

Arduino není jediná platforma na trhu, proto je vhodné mít povědomí i o jiných dostupných platformách jako například Raspberry Pi, PICAXE a Feather.

5.2.1 Raspberry Pi

Rspberry Pi jsou jednoduché mikro počítače. Od platformy Arduino se liší hlavně tím, že disponují vlastní operační systém, který těmto malým počítačům dává velký potenciál a díky tomu mají mnoho způsobů využití, jako jsou například centrální systémy správy domu nebo multimediální přehravače. Byly vyvinuty firmou Rastberry Pi Foundation. Jejich

prvotní účel byl jako vzdělávací pomůcka na školách, velice rychle se ale rozšířili i mezi veřejnost. [10]

Tyto mikropočítače disponují velkou rozmanitostí vstupních a výstupních portů, jako jsou USB, Ethernet nebo HDMI. [10]

Pro tento projekt jsou příliš komplikované a drahé.

5.2.2 PICAXE

Tyto mikroprocesory byly původně určeny ke vzdělávacímu využití na školách. Jedná se mikročip PIC, který je předprogramovaný PICKAXE bootstrapem. Vyznačuje se velice nízkou cenou a programují se v jazyce BASIC. [11]

Tento mikroprocesor je pro tento projekt příliš jednoduchý.

5.2.3 Feather

Mikroprocesory feather jsou narozdíl od mikroprocesoru ATmega které používá Arduino určené k řešení komplikovanějších úloh a vyznačují se větší pamětí. Některé jsou dokonce vybaveny WiFi čipem V celku je tato platforma velice podobna platformě Arduino Dokonce tyto mikroprocesory mohou být programovány ze stejného vývojového prostředí jako Arduino. Jediným velkým rozdílem je napětí. Na rozdíl od platformy Arduino kde se požívá 5 V, se zde používá napětí 3.3 V. To by mělo za následek, že k provozu určitých komponent jako je například čerpadlo by bylo zapotřebí používat externí zdroje napětí. [12]

5.2.4 Výhody platformy Arduino

Autor práce dopuručuje Arduino z několika důvodů.

- Má velice jednoduché a uživatelsky přívětivé IDE, se kterým se za krátký čas dokáže naučit pracovat i předem neznalý člověk, lze v něm však vytvořit i složitější projekty.
- Skutečnost že Wring vychází z jazyka C++, což je výhodou pro lidi, jež mají zkušenosti s programováním.
- Veliký sortiment jednotlivých komponentů, sensorů, čidel atp. + jejich příznivá cena a velká dostupnost.
- Je v komunitě nejrozšířenější. To má za následek, velkou podporu na fórech, spoustu dostupných knihoven a všeobecné know-how.

Z dostupných Arduino desek autor práce doporučuje Arduino Mega2560 z důvodu velkého množství pinů a tím i zvýšení pravděpodobnosti, že bude možné desku v budoucnu použít i na jiné projekty, aniž by bylo potřeba řešit problém s nedostatkem pinů.

5.3 Arduino IDE

Arduino IDE (Integrated Development Environment) je vývojové prostředí, ve kterém jsou tvořeny, kompilovány kódy a které jsou následně nahrány na desku. Je to v jistém smyslu textový editor obsahující několik funkcí. Má předinstalované ukázky kódu k práci s různými komponenty. Je zde i rozhraní pro sériovou komunikaci s počítačem. To znamená že deska může posílat informace do počítače, kde jsou následně zobrazovány, a naopak zvládá posílat informace z počítače zpět do běžícího programu. Nachází se zde i pomocník k instalaci přídavných knihoven od uživatelů. Umožňuje vytváření nových projektů, kompilaci vytvořeného kódu a po připojení desky k počítači i nahrání programu do mikroprocesoru přes připojený port. [13]

Obrázek 1 Arduino IDE

5.3.1 Wiring

Wiring je open-suorce programovací jazyk vycházející z jazyka C++ vytvořen pro programování mikrokontrolerů. Za tvůrce tohoto jazyka je považován Hernando Barragán.

Není určen výlučně pro Arduino. Používá se téměř ve všech případech ve spojení s ním a procesory ATmega. [14]

Podporuje používání knihoven, které usnadňují práci tím, že „schovávají“ části kódu řešící určitou úlohu v jednoduché příkazy. [15]

Jsou zde dostupné i globální proměnné, díky kterým je možné předávat si informace mezi funkcemi relativně jednoduše. Z pravidla bývají definovaná na samém začátku programu společně s knihovnami. [15]

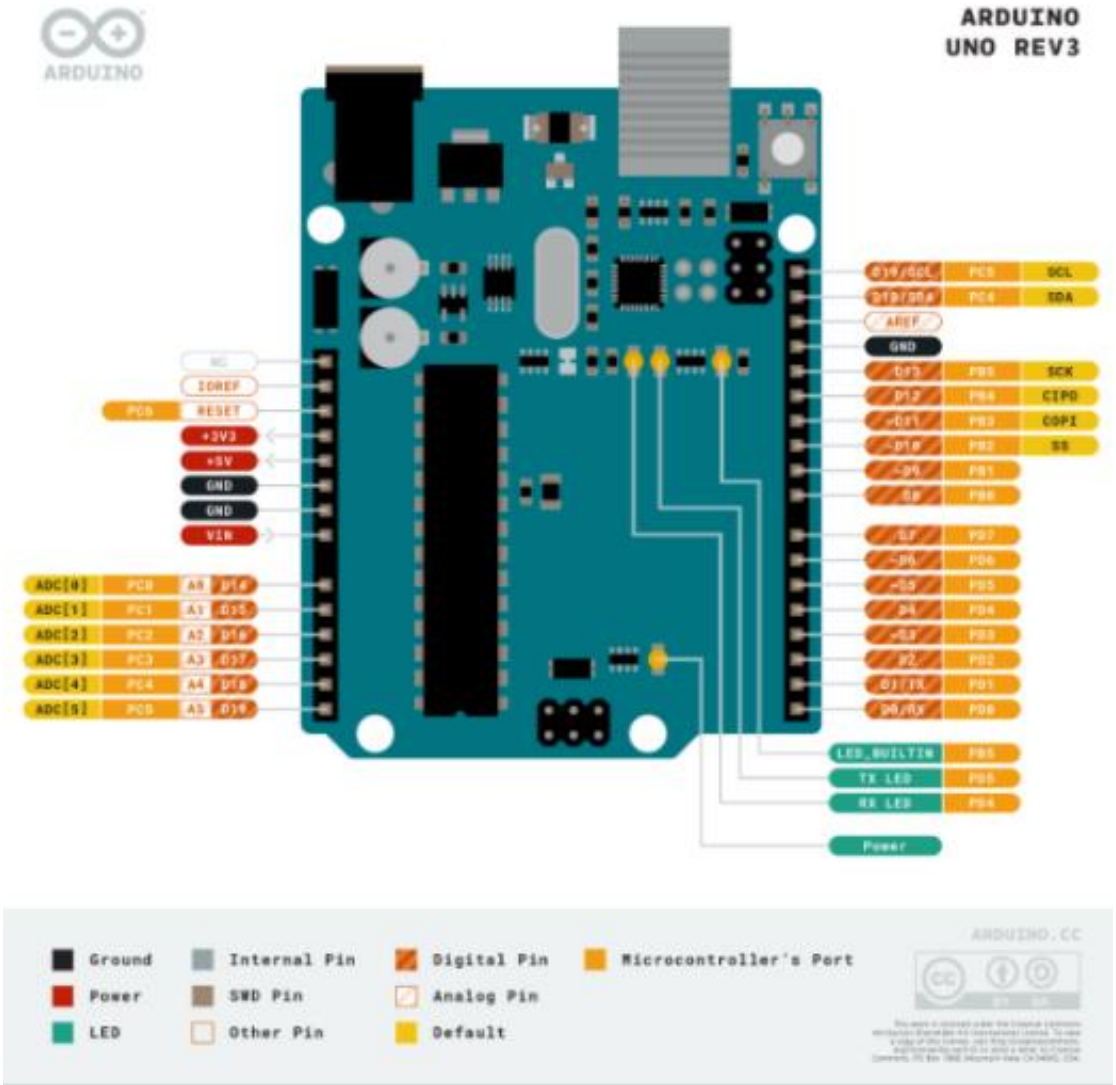
Bohužel Arduino desky mají jednoduché procesory bez operačního systému, který by podporoval paralelní průběhy kódů. Program je prováděn sekvenčně (postupně od shora dolu). [16]

Základní struktura Arduino programu jsou dvě funkce, „setup“ a „loop“. Setup proběhne jen při zapnutí/resetu desky a poté se přejde do loop, kde program zůstane v cyklu, dokud nebude deska vypnuta/resetována. [17]

Je zde možné používat i uživatelsky definované funkce, která mohou být definovány kdekoliv mimo setup a loop (před nimi, mezi nimi i za nimi). [18]

5.4 Arduino Uno Rev3

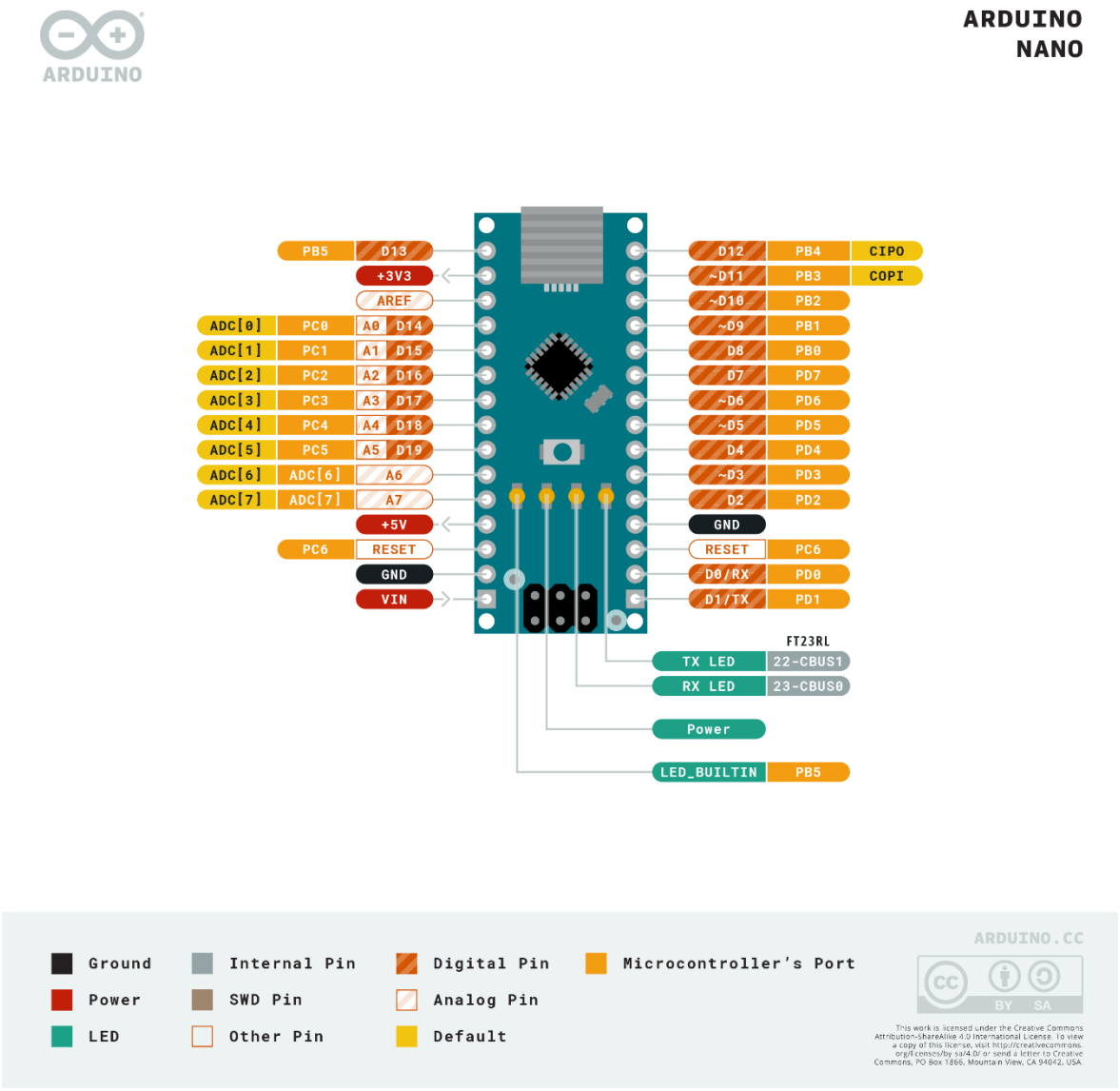
Arduino Uno Rev3 s mikroprocesorem ATmega328P je považována ze nejpoužívanější a nejrozšířenější deskou Arduino Jedná se o třetí generaci desek Uno. Na desce Arduino Uno Rev3 se nachází 14 digitálních vstupů a výstupů (6 z nich podporuje PWM) a 6 analogových vstupů, které mohou pracovat s 20 mA na jednotlivých pinech. Mikroprocesoru ATmega324 disponuje flash pamětí o velikosti 32 KB, a bootloader zabírá 0,5 KB. Arduino Uno Rev3 může být napájeno 6 až 20 V. Doporučené napětí je v rozmezí 7 až 12 V. [19]



Obrázek 2 Arduino UNO Rev3 [20]

5.5 Arduino Nano

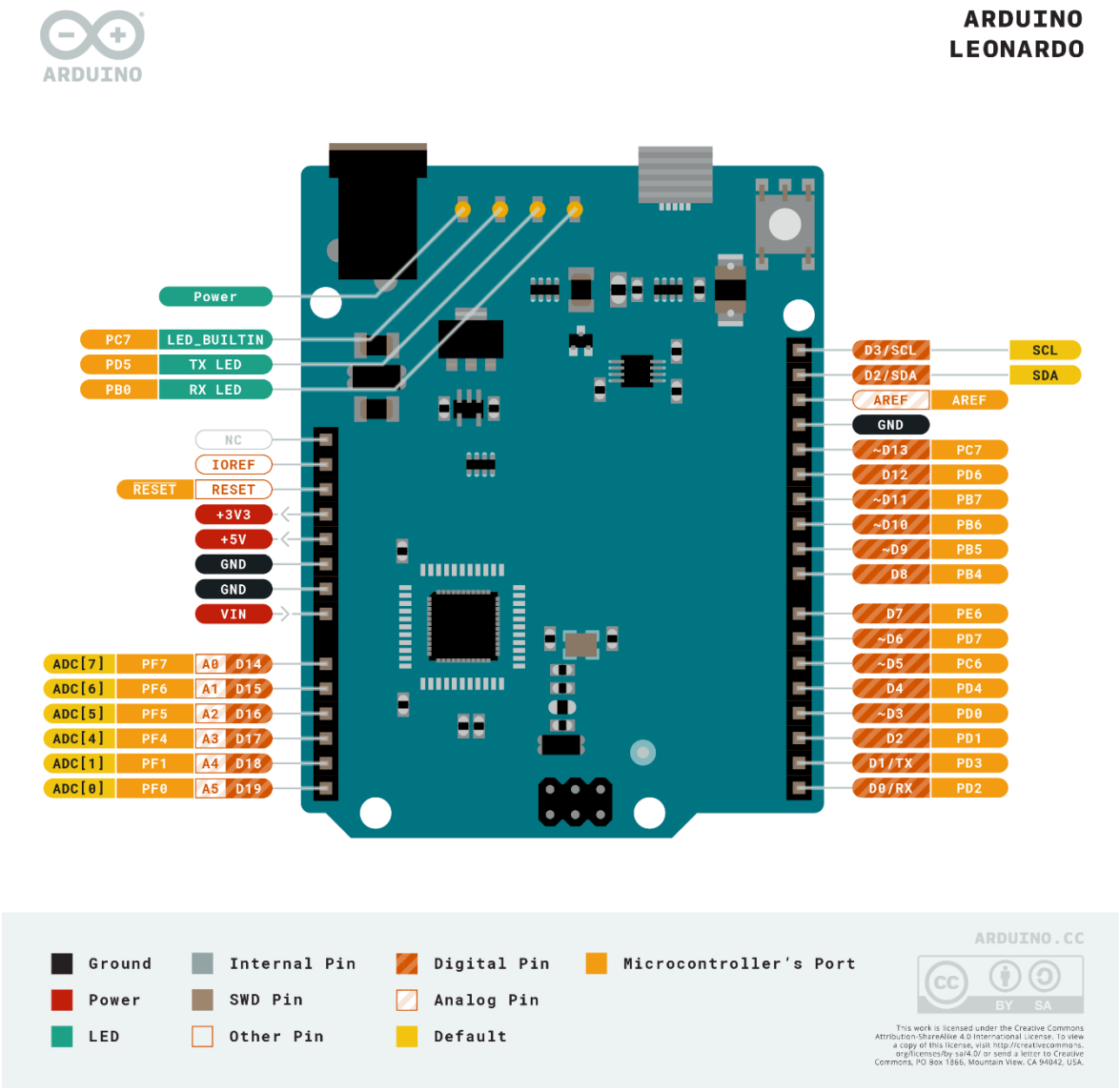
Arduino Nano s mikroprocesorem ATmega328 je nejmenší deskou z rodiny Arduino. I přes to ale víceméně nepřichází o žádnou funkcionalitu, které mají ostatní desky Arduino. Hlavním rozdílem je jeho velikost. Arduino Nano má dohromady 22 digitálních vstupů a výstupů (6 z nich podporuje PWM) a 8 analogových vstupů, které mohou pracovat s 40 mA na jednotlivých pinech. Paměť mikroprocesoru ATmega324 je omezena na 32 KB, kde bootloader zabírá 2 KB. Arduino Nano může být napájeno 6 až 20 V, doporučené je však rozmezí 7 až 12 V. [21]



Obrázek 3 Arduino Nano [22]

5.6 Arduino Leonardo

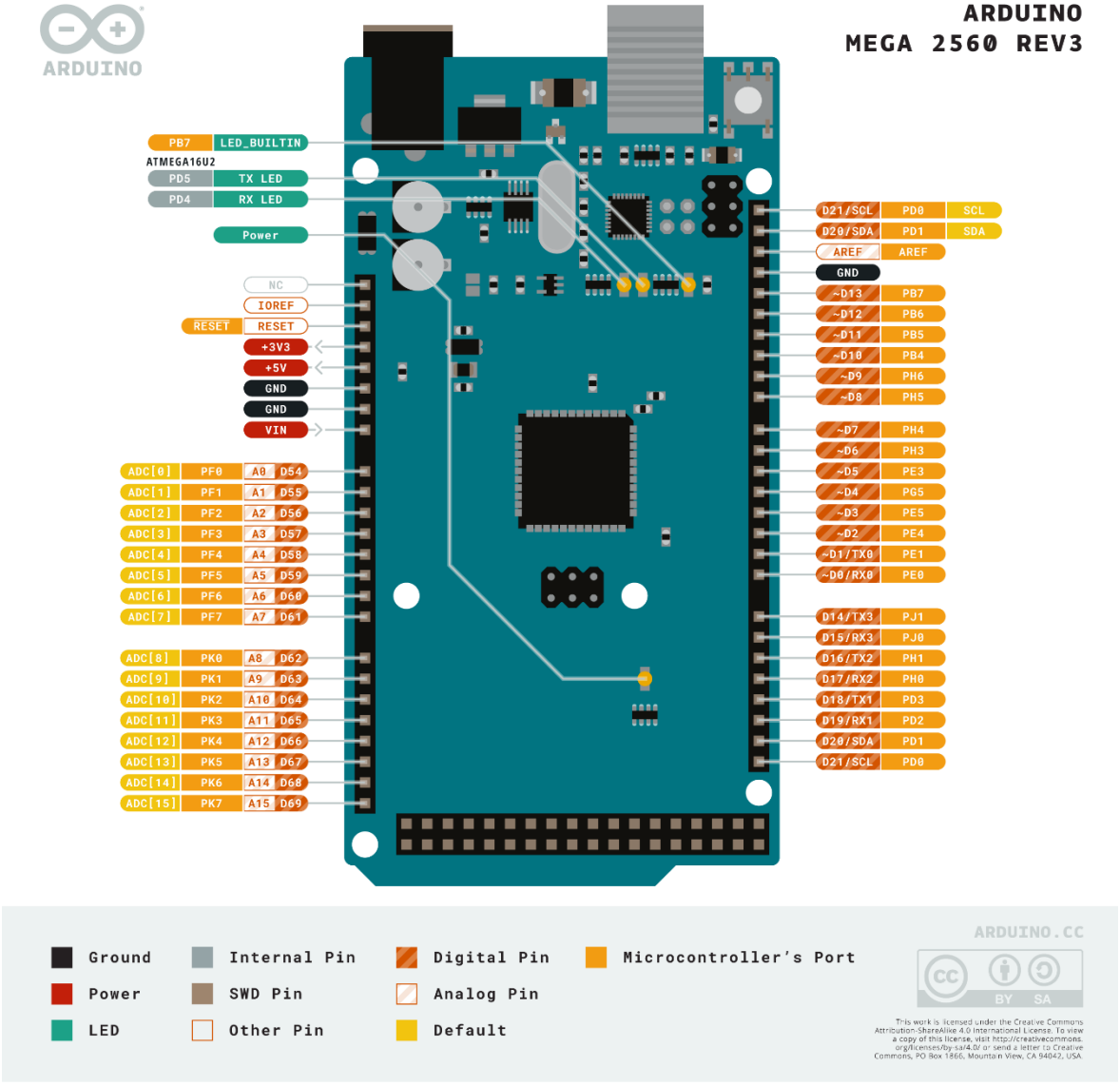
Arduino Leonardo s mikroprocesorem ATmega32u4 se od ostatních Arduin liší tím, že jeho mikrokontroler má vbudovanou USB komunikaci, proto nepotřebuje sekundární procesor. Arduino Leonardo má dohromady 20 digitálních vstupů a výstupů (7 z nich podporuje PWM výstup a 12 z nich muže být použito jako analgový vstup), které mohou pracovat s 40 mA na jednotlivých pinech. Paměť mikroprocesoru ATmega32u4 je omezena na 32 KB, kde bootloader zabírá 4 KB. Arduino Leonardo může být napájené 6 až 20 V, doporučené je ale rozmezí 7 až 12 V. [23]



Obrázek 4 Arduino Leonardo [24]

5.7 Arduino Mega 2560 Rev3

Arduino Mega2660 s mikroprocesorem ATmega2560 se vyznačuje svou velikostí. Jedná se zvětšenou verzi desky Arduino Uno Rev3 a jeho předchůdcem byla deska Arduino 1280. Arduino Mega2560 má dohromady 54 digitálních vstupů a výstupů (15 z nich podporuje PWM výstup) a 16 analgový vstupů, které mohou pracovat s 20 mA na jednotlivých pinech. Celkově se tedy na desce nachází 70 pracovních pinů. Paměť mikroprocesoru ATmega2560 je omezena na 256 KB, kde bootloader zabírá 8 KB. Arduino Mega2560 může být napájené 6 až 20 V, doporučené je ale rozmezí 7 až 12 V. [25]



Obrázek 5 Arduino Mega2560 [26]

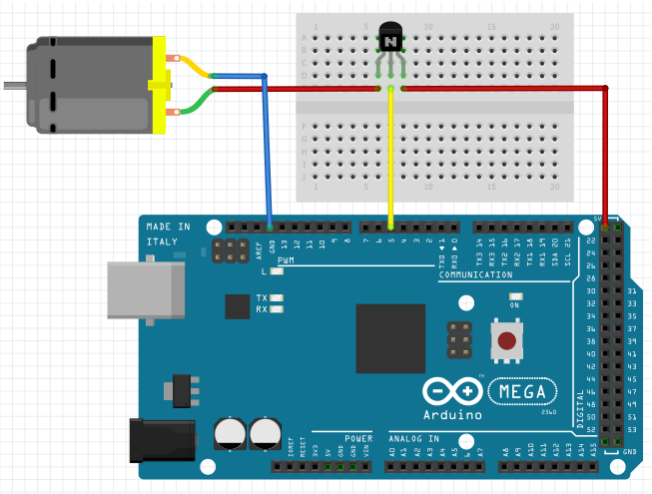
6 Použité komponenty

Tato kapitola je zaměřena na teoretickou část, kdy je čtenář seznámen s vybranými komponenty a jejich vlastnostmi, ze kterých se bude projekt následně skládat.

6.1 Ponorné mini čerpadlo

Toto čerpadlo je ve své podstatě elektromotor, v tomto případě pracující se stejnosměrným napětím o velikosti 3-6 V s proudovým odběrem 1 A, který jedním koncem vodu nasává a druhým ji žene ven. Průtok tohoto čerpadla se pohybuje mezi 80-120 l za hodinu a maximální zdvih se pohybuje v rozmezí 40-110 cm. Čerpadlo bude umístěno na dně nádržky na vodu. Autor vybral toto čerpadlo z důvodu, že na tento úkol je zcela dostačující, má příznivou cenu a pracuje s napětím 5 V. Ostatní čerpadla, na která autor narazil, potřebovala zdroj 12 V, což by značně komplikovalo práci. Při zapojení se autor rozhodl použít spínací NPN tranzistor ovládaný přes PWM pin, protože samotné PWM piny nebyly schopné čerpadlo uvést do provozu. [27]

Obrázek 6 Ponorné mini čerpadlo [28]

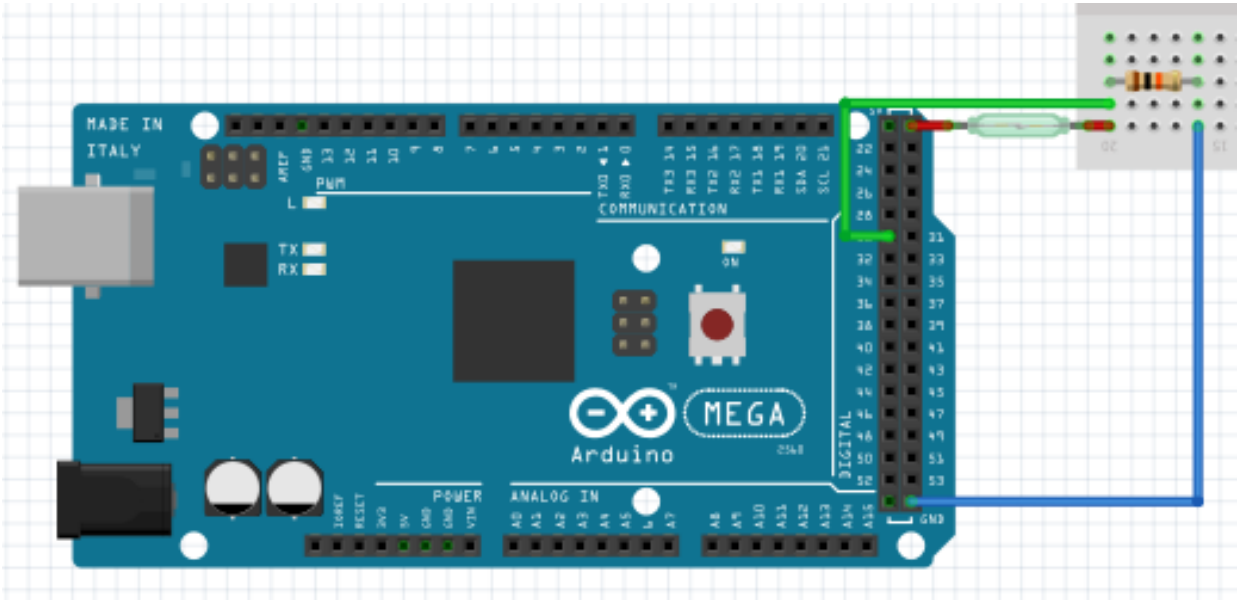


Obrázek 7 Zapojení čerpadla přes NPN tranzistor

6.2 Plovákový sensor vodní hladiny

Jedná se o velice jednoduchý spínací mechanismus. Skládá se z pevné a pohyblivé časti. V pevné části je umístěn jazýčkový kontakt citlivý na magnetické pole a v pohyblivé plovoucí části je magnet. Dostane-li se magnet do blízkosti kontaktu (tím, že klesne vodní hladina), dojde k sepnutí. Maximální spínací napětí je 100 V a proud 0,5 A. Při zapojení je použit i odpor o velikosti 10 000 ohm kvůli potlačení rušení. Pro toto čidlo se autor rozhodl zejména díky jeho jednoduchosti. [29]

Obrázek 8 Plovákový sensor hladiny [30]

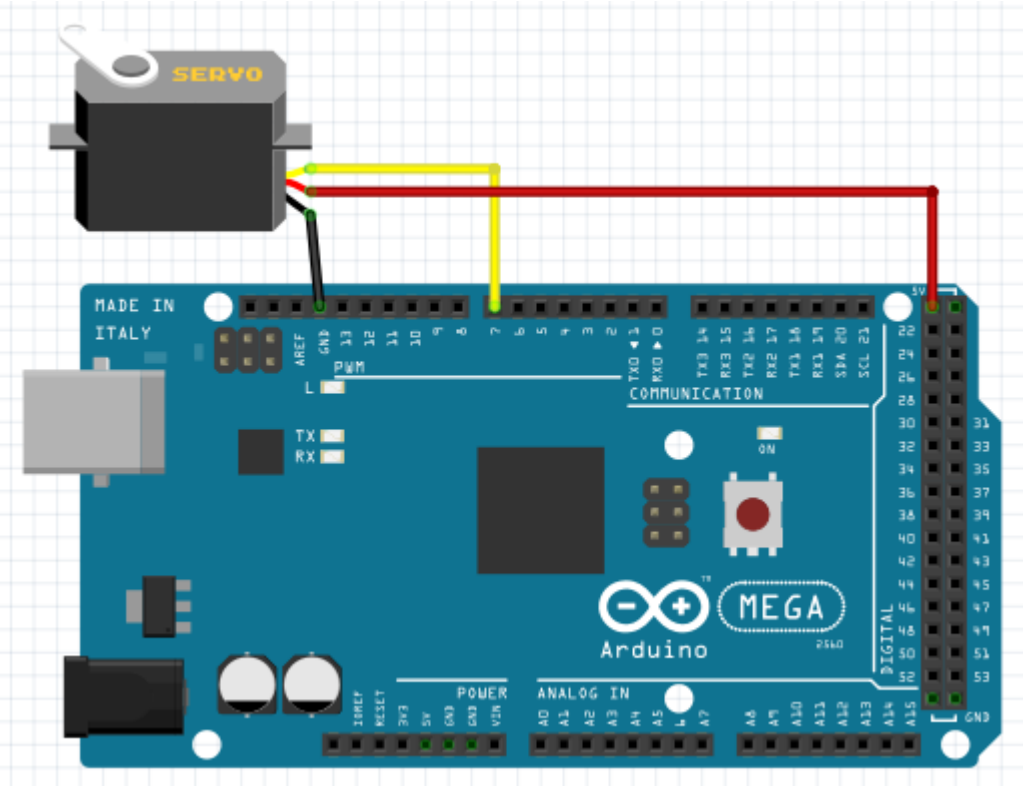


Obrázek 9 Zapojení plovákového sensoru s eliminací rušení

6.3 Servo MG90S

Toto servo disponuje kovovými převody. Ty zajišťují jeho dlouhodobější spolehlivost a přesnost oproti servům, jež mají převody plastové. Servo pracuje s napětím 4.8-6 V a provozní teploty jsou výrobcem definovány na 0-55 °C. Rotační úhel činí 180°. Rychlost serva závisí na vstupním napětí a to 0.1 s/60° při 4.8 V a 0.08 s/60° při 6.0 V. Točivý moment je taktéž závislí na napětí a pohybuje se 1,8 kg.cm při 4.8 V a 2.2 kg.cm při 6.0 V. Pro toto servo se autor rozhodl z důvodu kvality provedení, příznivé ceny, přesnosti a spolehlivosti. [31]

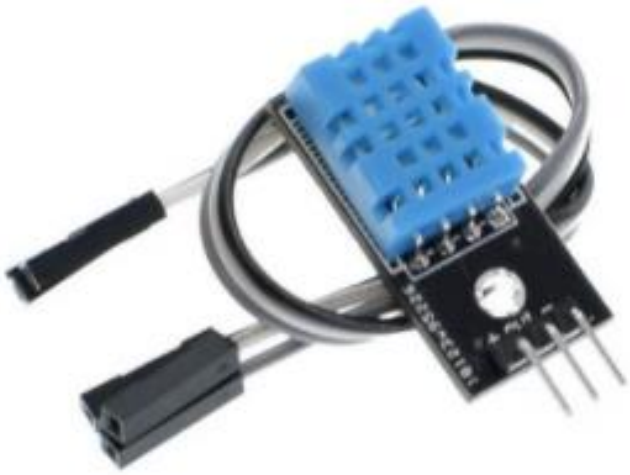
Obrázek 10 Servo MG90S [32]



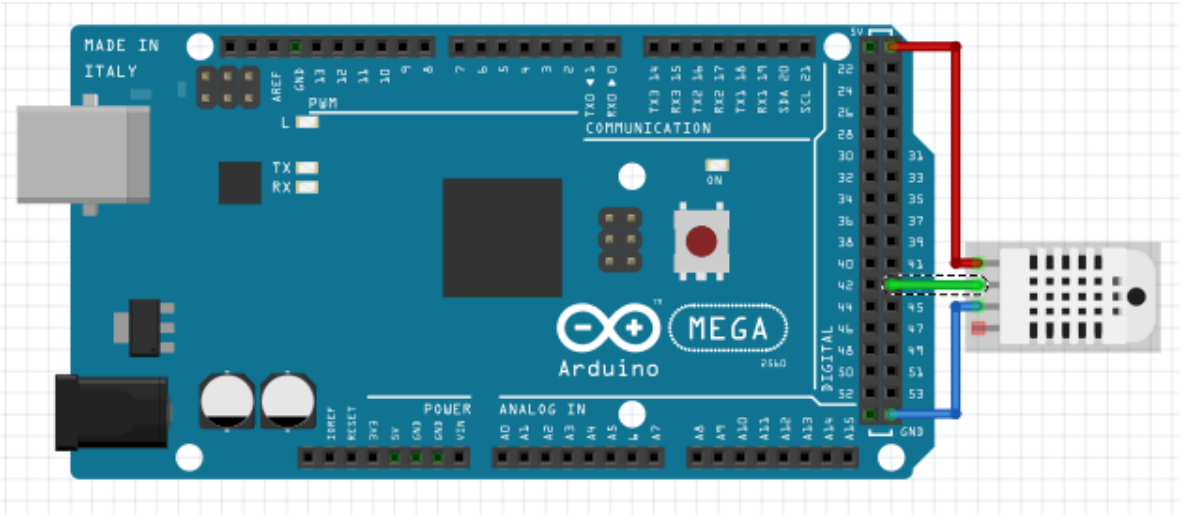
Obrázek 11 Zapojení serva.

6.4 Sensor teploty a vlhkosti vzduchu DHT11

Jde o jednoduchý sensor teploty a vlhkosti vzduchu. Modul má v sobě integrovaný pullup rezistor i kondenzátor, tudíž nejsou zapotřebí žádné další součástky k propojení čidla s deskou Arduino. Napájecí napětí se pohybuje v rozmezí 3.5-5 V. Rozsah měření se pohybuje v rozmezí 20-90 % s přesností 5 % u vlhkosti a 0-60 °C s přesností 2 °C u teploty. Toto čidlo je i přes velikosti odchylek pro projekt dostačující. [33]



Obrázek 12 Teplotní sensor DHT11 [34]

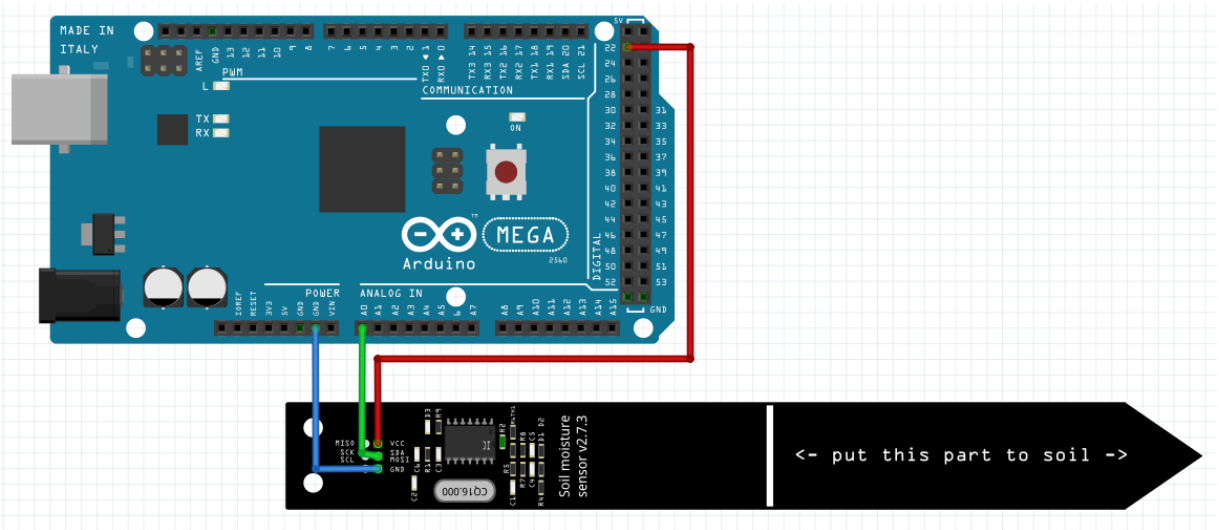


Obrázek 13 Zapojeni sensoru teploty

6.5 Kapacitní sensor vlhkosti půdy

Toto čidlo slouží k měření vlhkosti půdy. Napájecí napětí činí 5 V a odběr proudu se pohybuje kolem 5 mA. Naměřená hodnota se vrací na analogový výstup v rozmezí 0-3 V, odkud je odečítána hodnota na čidle, která je dále programově převedena na % hodnotu. Čidlo je pokryto antikorozním lakem, což prodlužuje jeho životnost. Při prvotním testování autor používal odporové čidlo, které funguje na jiném principu a u kterého velmi brzo nastal problém s korozí, proto se rozhodl vyměnit čidlo za kapacitní, které má znatelně delší životnost a odolnost vůči korozi. [35]

Obrázek 14 Kapacitní sensor vlhkosti půdy [36]

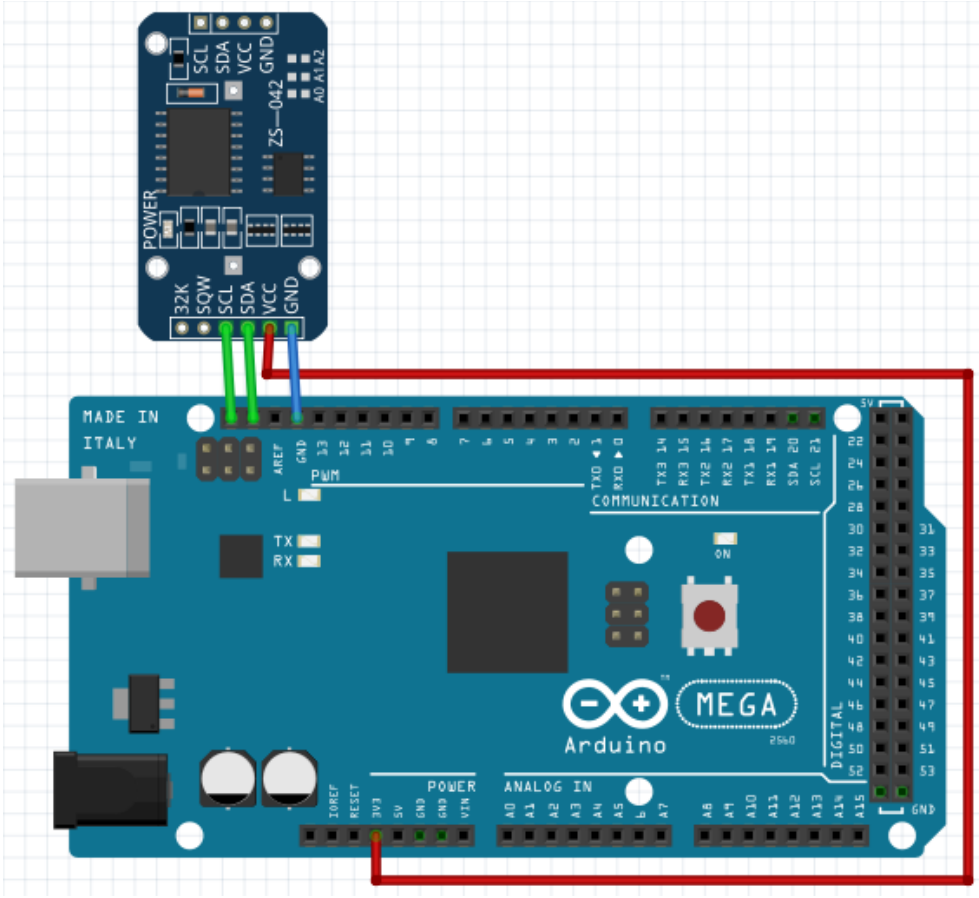


Obrázek 15 Zapojeni Kapacitního sensoru vlhkosti půdy

6.6 RTC Hodiny reálného času DS3231

Jedná se o relativně levné ale velice přesné hodiny reálného času. Dají se napájet jak 3.3 V tak 5 V a po přidání baterie jsou schopny uchovávat uložené informace. Skládají se ze dvou částí. První je DS3231 modul tvořen teplotně kompenzovaným krystalovým oscilátorem. Druhý je integrovaný paměťový modul AT24C32 typu EEPROM. Hodiny komunikují po sběrnici I2C s frekvencí 400 KHz. [37]

Obrázek 16 RTC Hodiny DS3231 [38]

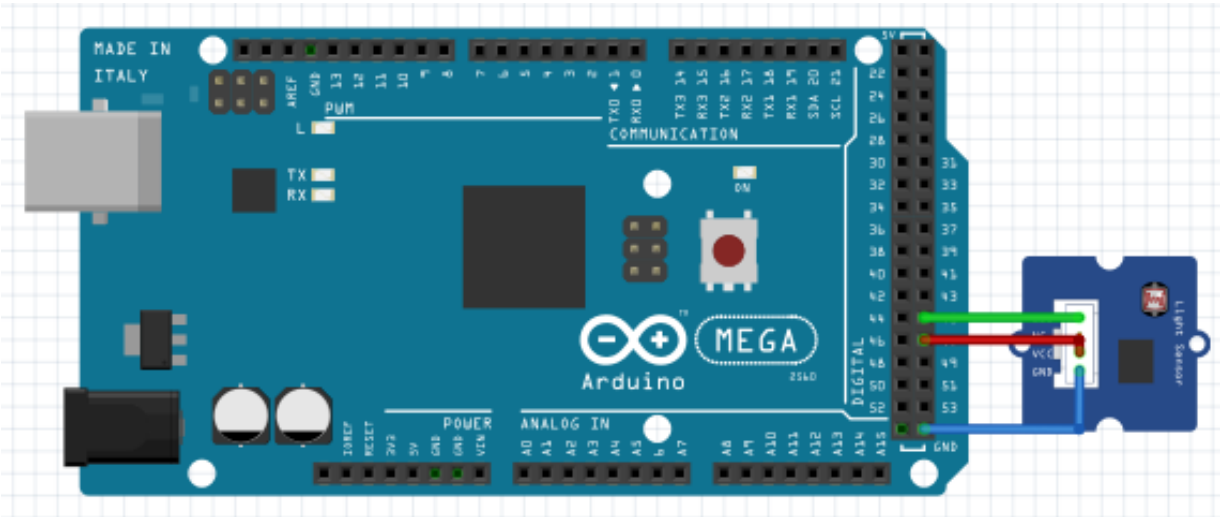


Obrázek 17 Zapojení RTC hodin

6.7 Světelný sensor

Jde o jednoduché čidlo pracující na principu fotorezistoru (světlo dopadající na fotorezistor mění jeho odpor, tato změna je následně detekována) a komparátoru LM393 (slouží k porovnání napětí a následnému vyhodnocení). Čidlo se nastavuje mechanicky přes jednoduchý trimr (mechanicky nastavitelný rezistor). Provozní napětí činí 3.3-5 V. Pro toto čidlo se autor rozhodl z důvodu jeho příznivé ceny a jednoduchosti. Malou nevýhodou je zmíněné ruční nastavení spínací hodnoty. Jde ale o zanedbatelný problém, protože autor práce nepočítá s častým měněním spínací hodnoty. [39]

Obrázek 18 Sensor světla [40]

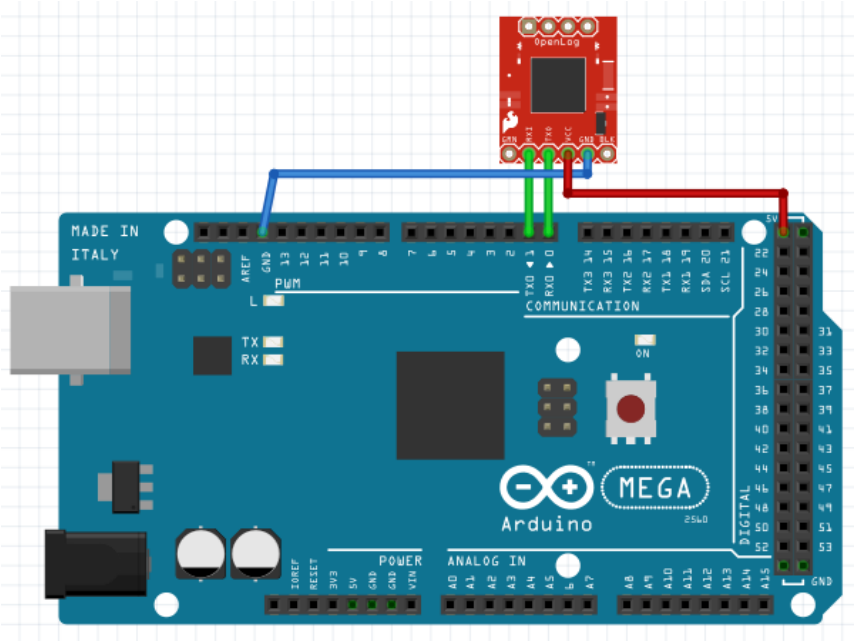


Obrázek 19 Zapojení sensoru světla

6.8 Displej Nextion

Autor se rozhodl pro Nextion orig. Enhanced NX3224K028 2.8" 320 x 240 TFT displej z důvodu, že na rozdíl od ostatních displejů nabízí dotykové ovládaní, což umožní vynechání mechanických tlačítek, spínačů, joysticku atp. a umožní ovládaní celého systému jen z displeje. Velikost displeje činí 2.8" a jeho rozlišení je 320 x 240. Obsahuje vlastní integrovaný MCU (mikrokontroler) s frekvencí 48 MHz. Obsahuje i 3 druhy pamětí, Flash o velikost 16 MB, SRAM o velikosti 3584 B a EEPROM o velikosti 1024 B. Používá vstupní napětí o velikosti 5 V a proud 500 mA. Provozní teploty se pohybují v rozemí -20-70 °C. Jde o velice kvalitní a pokročilý displej disponující mnoha funkcemi a možnostmi. Displej má k dispozici i vlastní počítačové rozhraní zvané NextionEditor, ve kterém probíhá veškeré nastavování, tvoření a modelovaní všeho, co bude na displeji zobrazováno. [41]

Obrázek 20 Displej Nextion 2.8“ [42]



Obrázek 21 Zapojení displeje

6.9 Další součástky

Ke správnému fungovaní nejsou potřeba jen samotná čidla ale i další komponenty a součástky, například led diody, které v tomto případě v projektu budou sloužit jako zdroj přídavného osvětlení nebo signalizace. [43]

Rezistory (odporové součástky snižující procházející proud například kvůli ochraně led diod před jejich spálením) [44]

Spínací NPN tranzistory (polovodičové součástky sloužící ke spínaní napětí přivedením malého napětí na bázi tranzistoru) [45]

Bzučák, který v tomto případě bude sloužit k signalizaci nedostatku vody přivedením napětí na jeho konektory. [46]

Nepájivá kontaktní pole, která budou sloužit k propojováni jednotlivých vodičů a komponent. [47]

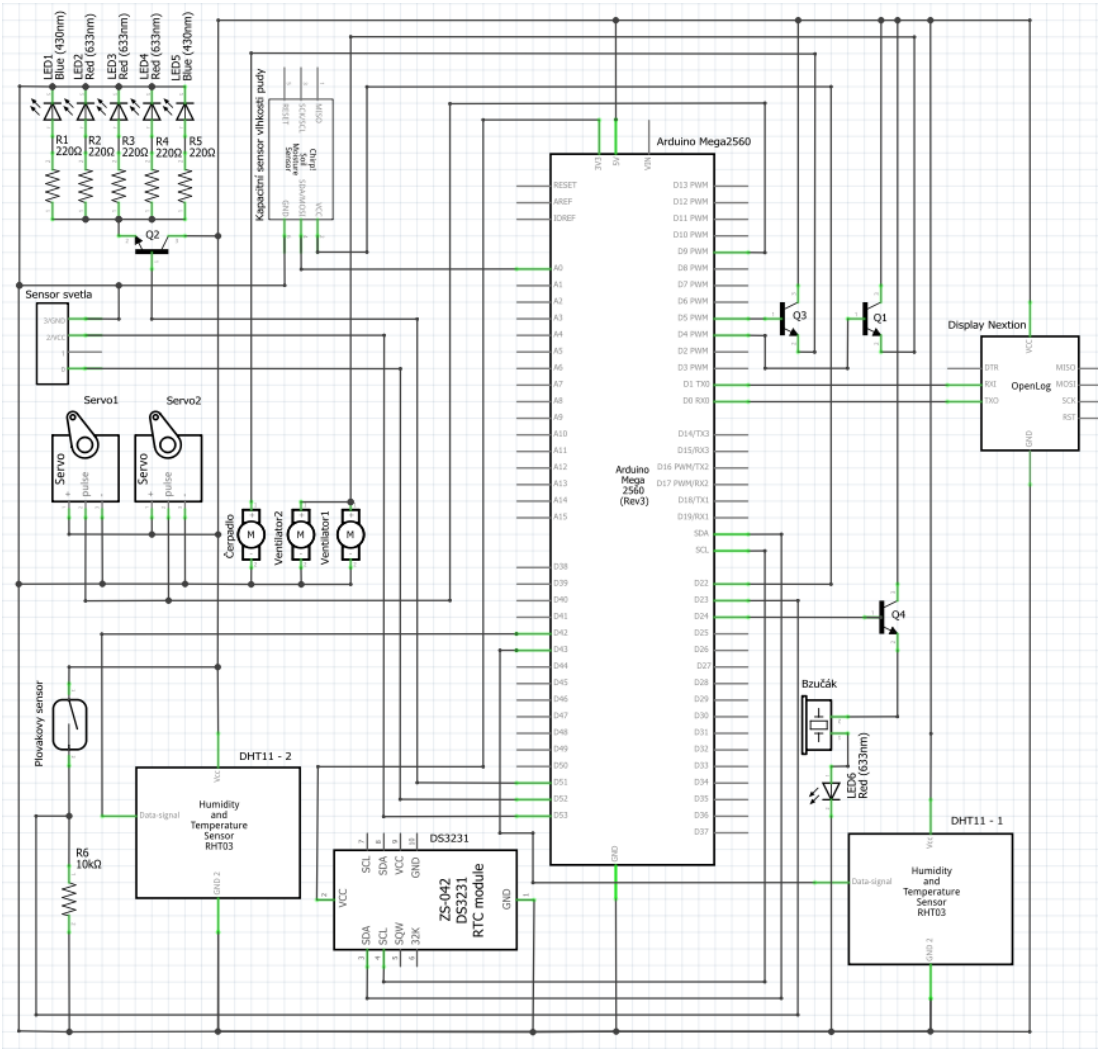
7 Zapojení projektu

Tato část práce se zabývá praktickým zapojením projektu, kde je navrženo elektrotechnické zapojení včetně grafického zapojení pro ukázku, jak bude reálně vypadat.

K zapojení jednotlivých komponentů autor požívá nepájivá kontaktní pole a vodiče k tomu určené. Autor zvolil tuto možnost pro její jednoduchost a možnost ručně měnit propojení. Další důvod je, že autor nemá k dispozici věci potřebné k pájení jako jsou například pájka, kalafuna, cín a vlastní vyleptaná spojová pole. Zapojení je v celku jednoduché, deska rozeznává vstupní a výstupní piny, jejichž pomocí komunikuje s komponenty.

7.1 Elektrotechnické schéma zapojení

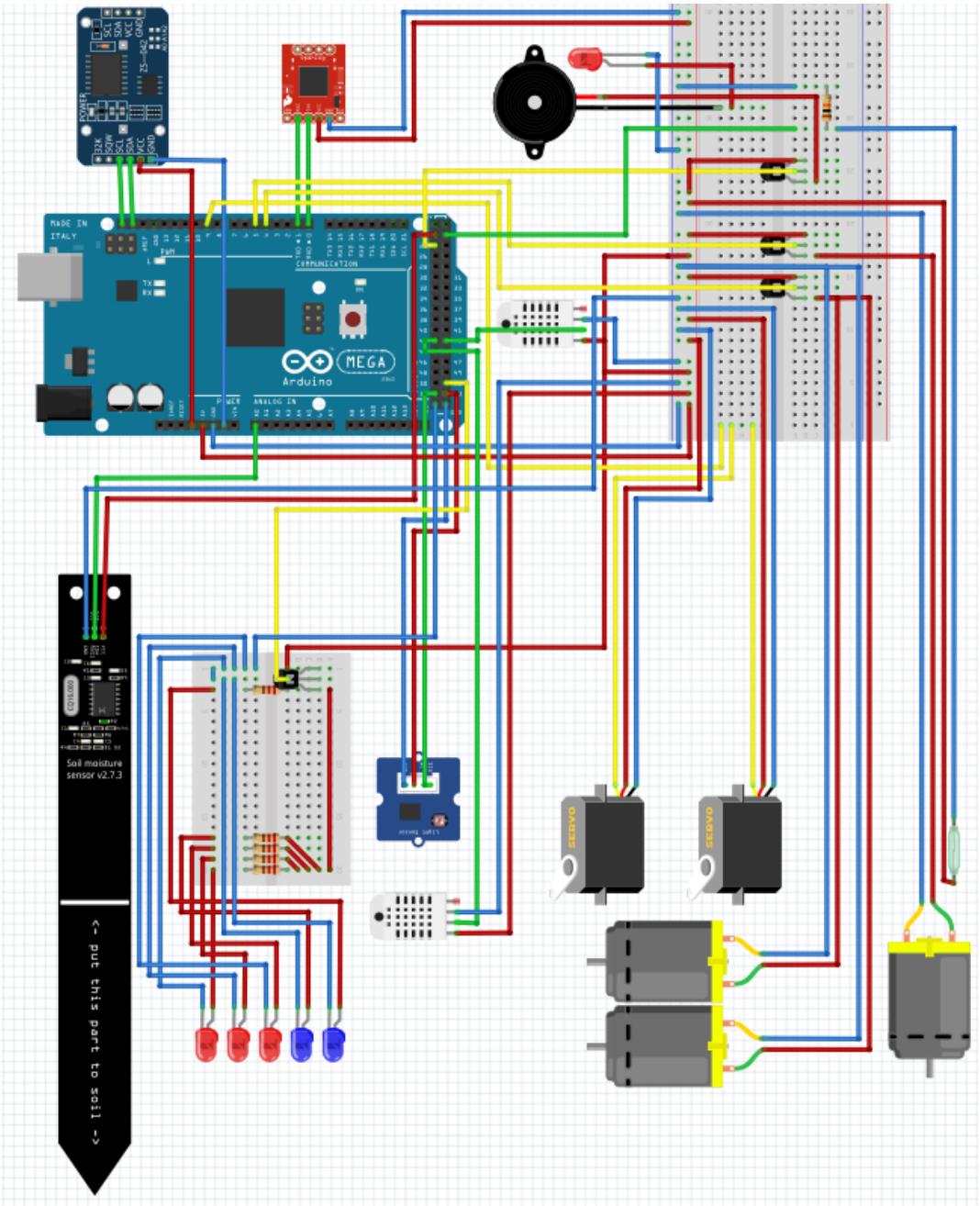
Elektrotechnické schéma zapojení slouží ke znázornění propojení všech použitých komponent, které jsou reprezentovány normovanými značkami.



Obrázek 22 Elektrotechnické schéma zapojení projektu

7.2 Grafické schéma zapojení

Grafické schéma zapojení slouží stejně jako elektrotechnické ke znázornění propojení všech použitých komponent se změnou, že komponenty jsou blízké svému skutečnému provedení z důvodu názornosti a přiblížení skutečnému vzhledu práce.



Obrázek 23 Grafické schéma zapojení projektu

8 Program

Tato kapitola je zaměřena na praktickou programovou část, kde je vysvětlen princip fungování programu platformy Arduino a displeje včetně důkladného popisu a vysvětlení použitých funkcí.

8.1 Funkční a nefunkční požadavky

Funkční požadavky

Program zajišťuje správu vstupních a výstupních periferií.

- Měření teplot
- Ovládání ventilace
- Měření vlhkosti půdy
- Kontrola vody v nádržce
- Spouštění čerpadla
- Měření času
- Kontrola stavu osvětlení
- Ovládání osvětlení

Nefunkční požadavky

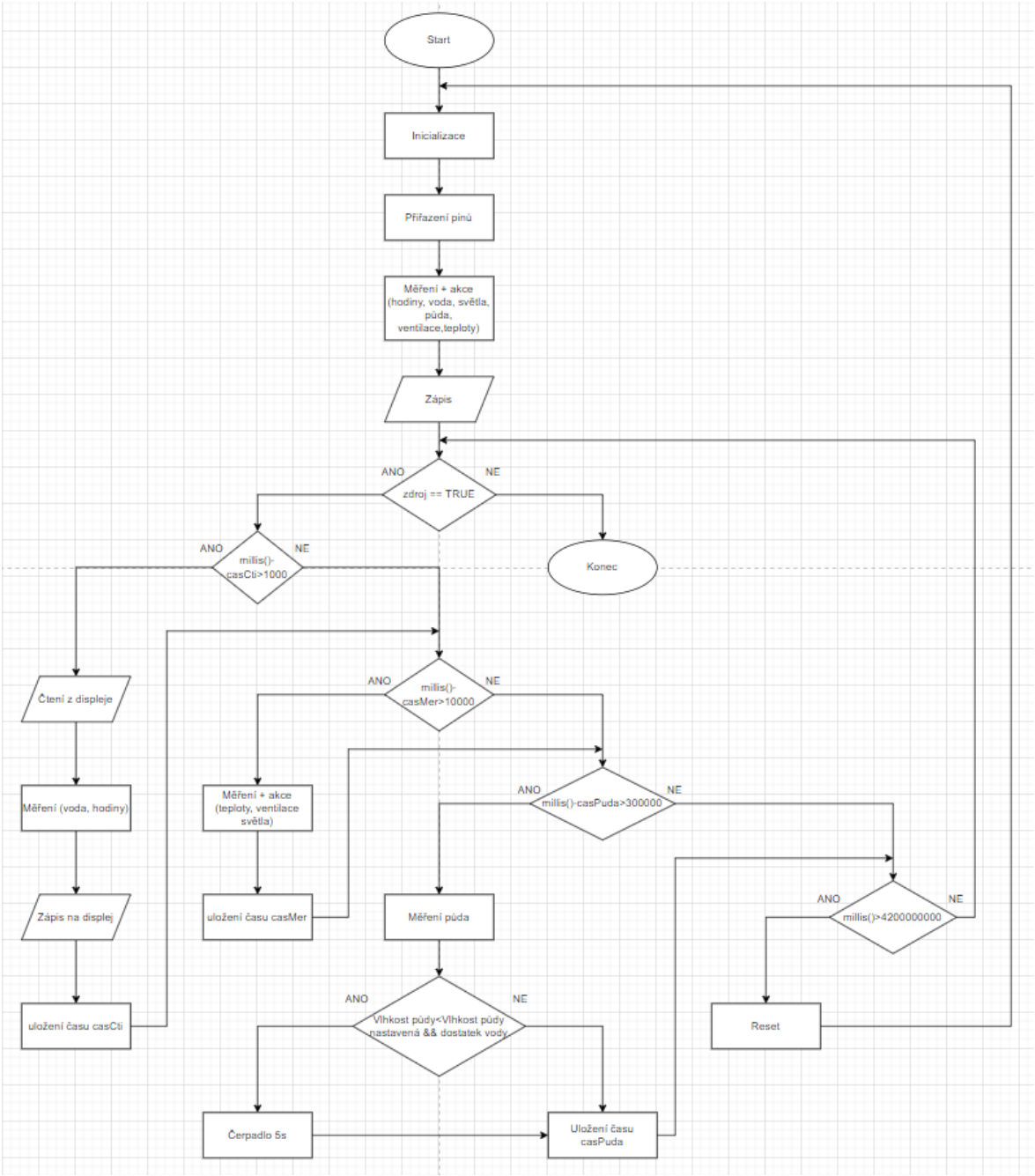
Program má následující vlastnosti

- Spolehlivost
- Přehlednost displeje a snadné ovládáni
- Rozšiřitelnost (to zajistí velká přehlednost díky použití uživatelských funkcí a vlastnost platformy Arduino snadného připojení nových komponent)
- Ochrana vlastních komponent (při vysokých teplotách skleníku by mohlo dojit k poškození sensorů).

Vše je důkladně rozepsáno a popsáno v dalších částech práce.

8.2 Vývojový diagram

Vývojový diagram slouží k jednoduchému popisu a znázornění jednotlivých kroků algoritmu.



Obrázek 24 Vývojový diagram algoritmu

8.3 Inicializace

Inicializace se nachází na úplném začátku programu. Jsou zde předdefinované knihovny, které budu v programu používat jako například DHT.h, což je knihovna pro komunikaci s teplotními sensory DHT11. DS3231.h je knihovna pro komunikaci

s hodinami, kde bylo zapotřebí použít i knihovnu Wire.h, kvůli komunikaci po SCL a SDA pinech. EasyNextionLibrary.h je knihovna pro komunikaci s displejem. Servo.h je, jak už název napovídá, knihovna pro práci se servy.

Dále se zde nachází definována čísla jednotlivých pinu pro následná použití v programu. Stejně tak proměnné, ve kterých budou později ukládána data.

Na závěr je zde uvedeno propojení sensorů teploty a serv., zahájení sériové komunikace s displejem, definování hodin a nastavení proměnné pro ukládaní hodnot z hodin.

```
#include "DHT.h"
#include "DS3231.h"
#include "EasyNextionLibrary.h"
#include <Servo.h>
#include <Wire.h>
#define Type DHT11
#define pudaAnalog A0
#define ventPin 4
#define cerpadloPin 5
#define servaPin 9
#define vccPinPuda 22
#define vodaPin 23
#define alarmPin 24
#define dhtPinIN 42
#define dhtPinOUT 43
#define svetloPin 51
#define svetloDigital 52
#define vccPinSvetlo 53
String strHodiny;
unsigned long casCti = 0;
unsigned long casMer = 0;
unsigned long casPuda = 0;
float teplotaIN;
float teplotaOUT;
int vlhkostPuda;
int teplotaMAX=20;
int vlhkostPMIN=50;
int PteplotaMAX=20;
int PvlhkostPMIN=50;
int voda;
int svetla;
int vent;
int modeVent=2;
int PmodeVent=2;
int modeSvetla=2;
int PmodeSvetla=2;
DHT IN(dhtPinIN,Type);
DHT OUT(dhtPinOUT,Type);
Servo serva;
EasyNex myNex(Serial);
DS3231 rtc;
RTCDateTime realCas;
```

Obrázek 25 Inicializace programu

8.4 Uživatelsky definované funkce

Uživatelsky definované funkce jsou funkce takové, které si programátor připravil sám a následně je používá v programu. Jejich hlavní výhodou je, že se tímto způsobem kód značně zpřehlední, zjednoduší a programátor se zbaví potřeby používat opakující se bloky kódu v hlavním programu.

Funkce servaVent

Jedná se o jednoduchou funkci na otevíráni a zavíraní serv. Funkce nemá žádnou návratovou hodnotu, proto zde uvedeno před funkcí void. Funkce má jeden vstupní parametr, a to celočíselnou hodnotu úhlu na kolik se mají serva natočit.

Tato funkce funguje následovně, nejprve si přečte, v jaké poloze se serva nacházejí, a podle toho se rozhodne, zda bude serva otevírat či zavírat. Samotný pohyb je uskutečněn v cyklu for, kde se pomalu mění hodnota proměnné i a tato hodnota se zapisuje na PWM výstupní pin serv. Je zde použit i krátký delay(5), aby se serva nepohybovala příliš rychle a tím se zamezilo případnému cukání.

```
void servaVent(int uhelCil)
{
  int uhel = serva.read();
  if (uhelCil > uhel)
  {
    for (int i = uhel; i < uhelCil ; i++)
    {
      serva.write(i);
      delay(5);
    }
  }
  else
  {
    for (int i = uhel; i > uhelCil; i--)
    {
      serva.write(i);
      delay(5);
    }
  }
}
```

Obrázek 26 Funkce na pohyb serv

Funkce merPuda

Úkolem této funkce je změřit hodnotu vlhkosti půdy a tuto informaci poslat dál. Nejsou zde žádné vstupní parametry, se kterými by funkce pracovala. Funkce vrací

celočíselnou hodnotu reprezentující procentuální hodnotu vlhkosti od 0 % do 100 % ve formátu int.

Na napájecí pin sensoru se přivede 5 V. Počká se 100 ms z důvodu ustálení napětí na sensoru. Dále se přečte analogová hodnota, která se uloží do proměnné, se kterou budeme dále pracovat. Poté už není potřeba napájet sensor, tudíž ho vypneme vypnutím napětí na napájecím pinu. Uložená hodnota se musí zpracovat, proto je použita funkce map(), která namapuje analogovou hodnotu na procentuální. Hodnoty mapování byly nastaveny podle kalibrace čidla, která byla provedena při konstrukci. V tomto případě na vzduchu mimo půdu, což odpovídá hodnotě vlhkosti půdy 0 %, byla naměřena hodnota 800 a při ponoření čidla do vody, což odpovídá hodnotě 100 % vlhkosti, byla naměřena hodnota 425. Po úspěšném zkalibrování a namapování je zapotřebí vyřešit problém s odchylkami. O to se postará jednoduchá podmínka, která všechny hodnoty menší než 0, převede na 0 a hodnoty vetší než 100 převede na 100.

```
int merPuda()
{
  digitalWrite(vccPinPuda, HIGH);
  delay(100); //ustaleni napeti
  int vpuda = analogRead(pudaAnalog);
  digitalWrite(vccPinPuda, LOW);
  vpuda = map(vpuda, 425, 800, 100, 0); //upravení analogové hodnoty od 0% do 100%
    if (vpuda < 0)
    {
      vpuda = 0;
    }
    if (vpuda > 100)
    {
      vpuda = 100;
    }
  return vpuda;
}
```

Obrázek 27 Funkce na měření vlhkosti půdy

Funkce cerpadlo

Jednoduchá funkce na spuštění čerpadla na určitou dobu a jeho následné vypnutí. Funkce nemá žádnou návratovou hodnotu, proto před ní uveden void. Má jeden vstupní parametr, což je celočíselná hodnota reprezentující kolik milisekund má čerpadlo spušťěné

Princip je jednoduchý, na pin ovládající čerpadlo je přivedena logická 1 (5 V). Poté se vykoná zdržení funkcí delay() po dobu, kterou známe e vstupního parametru. Po uplynutí zdržení se na pin čerpadla přivede logická 0 (0 V) a čerpadlo se vypne.

```
void cerpadlo(int casCerpadlo)
{
  digitalWrite(cerpadloPin, HIGH);
  delay(casCerpadlo);
  digitalWrite(cerpadloPin, LOW);
}
```

Obrázek 28 Funkce na spouštění čerpadla

Funkce ventilace

Jde o funkci, která se stará o spouštění a vypínaní ventilace pro účely chlazení, respektive o otevírání průduchů a spouštění ventilátorů. Funkce má několik vstupních parametrů, a to aktuální teplotu ve skleníku, teplotu mimo skleník a ideální teplotu. Vrací se informace, jestli je ventilace spuštěna či nikoliv.

Principiálně jde jen o jednu podmínku, kde se kontroluje, zda je teplota uvnitř větší než ideální teplota, jinak by nebylo odůvodnění větrat. Zda není ručně nastaveno vypnutí ventilace a zda je teplota uvnitř vyšší než teplota venku aspoň o 2 °C. To z důvodu nepřesnosti čidel. Postačí, když je nastavené ruční spuštění ventilace uživatelem z ovládacího displeje. V ten moment se otevřou serva a spustí se ventilátory přivedením napětí na jejich ovládací pin. V opačném případě se ventilátory zastaví vypnutím napětí na jejich ovládacím pinu a serva uzavřou průduchy. Následně funkce vrátí informaci o stavu ventilace.

```
int ventilace(float teplotaIN, float teplotaMIN, int teplotaMAX)
{
  if((teplotaIN > teplotaMAX && modeVent != 0 && teplotaMIN+2 < teplotaIN) || (modeVent==1))
  {
    servaVent(100);
    digitalWrite(ventPin, HIGH);
    return 1;
  }
  else
  {
    digitalWrite(ventPin, LOW);
    servaVent(0);
    return 0;
  }
}
```

Obrázek 29 Funkce na spouštění ventilace

Funkce cti

Funkce slouží k přečtení nastavení z ovládacího displeje. Nejsou zde žádné vstupní parametry a nevracíme ani žádnou hodnotu.

Do proměnné teplotaMAX symbolizující ideální teplotu se pomocí funkce z knihovny pro práci s displejem načte nastavená hodnota. Při testování autor narazil na problém s nezdařeným čtením, proto bylo zapotřebí ošetření. Při nezdařeném čtení se načte hodnota -8655. Nenastane-li tato situace, počítá se s tím, že se čtení povedlo a do pomocné proměnné, která si pamatuje poslední nastavenou hodnotu, se zapíše hodnota nová. Nastane-li problém se čtením, respektive načte-li se nevalidní hodnota -8655, přepíše se posledním nastavením a při dalším cyklu čtení by se měla už načíst správně. Program čte z displeje každou vteřinu, tudíž jde o zanedbatelnou ztrátu. Poté se kontroluje relevantnost nastavení. Ve specifikacích pro čidla byly definovány funkční hodnoty od 0 do 55 °C, proto jsou nastaveny jako hranice u podmínek. Přesáhne-li nastavená hodnota tyto hranice, automaticky se na displeji přepíše na okrajové funkční hodnoty (-60 stupňů se přepíše na 0, 120 stupňů se přepíše na 55).

To stejné proběhne u čtení nastavení ideální vlhkosti půdy, jen se změnami, že limitní hodnoty jsou zde nastaveny na 0-90 % (90 z důvodu, že dosáhnout vyšších hodnot vlhkosti v půdě by mohlo mít za následek vytopení)

```
if (teplotaMAX != -8655)
{
  PteplotaMAX = teplotaMAX;
}
else if (teplotaMAX == -8655)
{
  teplotaMAX = PteplotaMAX;
}
if(teplotaMAX > 55)
{
  teplotaMAX = 55;
  myNex.writeNum("nteplota.val", teplotaMAX);
}
else if(teplotaMAX < 0)
{
  teplotaMAX = 0;
  myNex.writeNum("nteplota.val", teplotaMAX);
}
vlhkostPMIN = myNex.readNumber("nvlhkost.val");
if (vlhkostPMIN != -8655)
{
  PvlhkostPMIN = vlhkostPMIN;
}
else if (vlhkostPMIN == -8655)
{
  vlhkostPMIN = vlhkostPMIN;
}
if(vlhkostPMIN > 90)
{
  vlhkostPMIN = 90;
  myNex.writeNum("nvlhkost.val", vlhkostPMIN);
}
else if(vlhkostPMIN < 0)
{
  vlhkostPMIN = 0;
  myNex.writeNum("nvlhkost.val", vlhkostPMIN);
}
```

Obrázek 30 První část funkce čtení z displeje

U čtení módu ventilace a světel se pouze kontroluje, zda se hodnoty povedlo načíst, zde se nemohou ručně nastavovat jiné hodnoty než 0 -vypnuto, 1 -zapnuto, 2 -automatický režim řízen mikrokontrolerem. Není tedy potřeba kontrolovat tyto hodnoty zpětně.

```
  modeVent = myNex.readNumber("nvent.val");
  if (modeVent != -8655)
  {
    PmodeVent = modeVent;
  }
  else if (modeVent == -8655)
  {
    modeVent = PmodeVent;
  }
  modeSvetla = myNex.readNumber("nsvetla.val");
  if (modeSvetla != -8655)
  {
    PmodeSvetla = modeSvetla;
  }
  else if (modeSvetla == -8655)
  {
    modeSvetla = PmodeSvetla;
  }
}
```

Obrázek 31 Druhá část funkce na čtení z displeje

Funkce zapis

Jedná se o funkci, která zapisuje získané informace pro uživatele a ovládací displej. Funkce nevrací žádnou hodnotu, za to má značné množství vstupních parametrů. Jsou to čas (informace o čase ve formátu hh:mm), světla (informace, zda jsou světla zapnuta či vypnuta reprezentovaná 0/1), vent (informace, zda je spuštěna ventilace reprezentována 0/1), tIN (informace o tom, jaká je teplota uvnitř skleníku), tOUT (informace o teplotě mimo skleník), voda (informace, zda je stav vody v pořádku reprezentován hodnotou 0/1) a vPuda (informace o vlhkosti půdy).

Informace o čase se uloží do textového řetězce, který se později odešle na displej. Textový řetězec pro světla je tvořen slovem „Světla“ a dle podmínky je k němu přiřazeno ON nebo OFF dle hodnoty vstupního parametru. To stejné proběhne i s řetězcem pro ventilaci. Dále se vytvoří textové řetězce pro teplotu zřetězením“Teplota uvnitř/venku: “ a hodnotou ze vstupního parametru + znak “ C”. Velice podobně se zapíše i informace o vlhkosti půdy, jen se změnou, že jednotky jsou uvedeny v %. Na závěr se vytvoří řetězec pro informaci o vodě rozhodnutím v podmínce dle vstupního parametru pro vodu.

Všechny výsledné řetězce s informacemi se odešlou na displej příkazem z knihovny pro práci s displejem a tím dojde k jejich zapsání.

```
void zapis(String cas, int svetla, int vent, float tIN, float tOUT, int voda, int vPuda)
{
  String Dcas = cas;
  String Dsvetla = "Svetla";
  if (svetla == 0)
  {
    Dsvetla += " OFF";
  }
  else
  {
    Dsvetla += " ON";
  }
  String Dvent = "Ventilace";
  if (vent == 0)
  {
    Dvent = "Ventilace OFF";
  }
  else
  {
    Dvent = "Ventilace ON";
  }
  String DtIN = "Teplota uvnitr: ";
  DtIN += tIN;
  DtIN += " C";
  String DtOUT = "Teplota venku: ";
  DtOUT += tOUT;
  DtOUT += " C";
  String DvPuda = "Vlhkost pudy: ";
  DvPuda += vPuda;
  DvPuda += " %";
  String Dvoda = "";
  if (voda == 0)
  {
    Dvoda += "Voda OK";
  }
  else
  {
    Dvoda = "Dolij vodu";
  }
  myNex.writeStr("thodiny.txt", Dcas);
  myNex.writeStr("tsvetla.txt", Dsvetla);
  myNex.writeStr("tventilace.txt", Dvent);
  myNex.writeStr("tteplotain.txt", DtIN);
  myNex.writeStr("tteplotaout.txt", DtOUT);
  myNex.writeStr("tvoda.txt", Dvoda);
  myNex.writeStr("tvlhkostpuda.txt", DvPuda);
}
```

Obrázek 32 Funkce pro zápis dat na displej

Funkce testVoda

Úkolem této funkce je kontrolovat a popřípadě signalizovat nedostatek vody v nádržce na vodu. Funkce vrací hodnotu udávající, zda je vody dostatek či nikoliv. Nejsou zde žádné vstupní parametry.

Plovákové čidlo je připojené na stálých 5 V a kontroluje se, zda je sepnuté či nikoliv. V případě, že je sepnuté, aktivuje se signalizační dioda a bzučák. Je zde nastaveno, aby se signalizace zapnula, pokud byla předtím vypnuta, a aby se v opačném případě vypnula. Tato funkce je později použita v cyklu, jenž probíhá každou vteřinu. To způsobí blikání signalizační diody a přerušované houkání bzučáku, které upozorní uživatele na nedostatek vody. Autor zvolil přerušované blikání a houkaní z důvodu, že přerušovaná signalizace upoutá pozornost (uživatele) spíše než statická. V případě, že je vody dostatek a plovákové čidlo není sepnuté, vypne se signalizace byla-li předtím zapnuta.

```
int testVoda()
{
  if(digitalRead(vodaPin) == HIGH)
  {
    digitalWrite(alarmPin, !digitalRead(alarmPin));
    return 1;
  }
  else
  {
    digitalWrite(alarmPin, LOW);
    return 0;
  }
}
```

Obrázek 33 Funkce na kontrolu dostatku vody v nádrži

Funkce ctiCas

Jde o funkci, která vrací hodnotu času v podobě stringu (textového řetězce) ve formátu hh:mm. Používá se pouze pro zobrazování času na ovládacím displeji.

Vytvoří se prázdný string, ve kterém bude později hodnota času. Do pomocné proměnné se uloží informace o čase (datum, čas, rok) program si z nich následně vybere hodiny a minuty, zbytek informací o čase je v tuto chvíli irelevantní. U minut nastává problém, když jsou jednociferné, protože se před nimi nevypisuje 0, tudíž by výsledný čas mohl vypadat takto: 14:1 namísto 14:01. To zabezpečuje podmínka, která když zjistí, že číslo reprezentující minuty je jednociferné, přidá před něj znak „0“. Následně funkce vrací hodnotu času ve formátu, jaký jsme požadovali.

```
String ctiCas()
{
  String strHodinyPom = "";
  realCas = rtc.getDateTime();
  strHodinyPom += realCas.hour;
  strHodinyPom += ":";
  if(realCas.minute < 10)
  {
    strHodinyPom += "0";
    strHodinyPom += realCas.minute;
  }
  else
  {
  strHodinyPom += realCas.minute;
  }
  return strHodinyPom;
}
```

Obrázek 34 Funkce na přečtení času z hodin

Funkce svetlo

Tato funkce slouží ke kontrole vnějšího osvětlení a případného rozsvícení nebo zhasnutí přídavného světla. Funkce vrací hodnotu signalizující, zda jsou světla rozsvícena nebo zhasnuta, proto je před funkcí použit int, který pro tento případ postačí. Nejsou zde použity žádné vstupní parametry.

Na pin sensoru světla je přivedena logická 1 (5 V). Poté se vyčká 100 ms pro ustálení napětí. Dále se přečte aktuální hodnota z hodin a uloží se do proměnné, aby mohla být následně použita u rozhodováni, zda světla rozsvítit nebo zhasnout. Pro rozsvícení musí být splněny určité podmínky. Musí být zjištěn nedostatek světla, zároveň nesmí být světla nastavena na vypnuto a musí být určitý čas, který je napevno nastaven od 7 do 19 (to z důvodu, aby se nesvítilo v noci, kdy je zapotřebí aby byla tma). Stačí, aby bylo osvětlení zapnuto uživatelem z ovládacího displeje. Když se rozhodne, jestli má být rozsvíceno nebo zhasnuto, tzn. jestli se na pin ovládající světla přivede napětí či ne, vypne se napájení sensoru světla a vrátí se hodnota reprezentující, zda jsou světla rozsvícena nebo zhasnuta.

```
int svetlo()
{
  digitalWrite(vccPinSvetlo, HIGH);
  delay(100);
  realCas = rtc.getDateTime();
  if(((digitalRead(svetloDigital) == HIGH) && modeSvetla != 0 && realCas.hour > 7 && realCas.hour < 19) || (modeSvetla==1))
  {
    digitalWrite(svetloPin, HIGH);
    digitalWrite(vccPinSvetlo, LOW);
    return 1;
  }
  else
  {
    digitalWrite(svetloPin, LOW);
    digitalWrite(vccPinSvetlo, LOW);
    return 0;
  }
}
```

Obrázek 35 Funkce na spouštění světel

Funkce reset

Funkce reset slouží k resetování systému. Je zapotřebí z důvodu rizika přetečení proměnných pro ukládání časů. Tato funkce byla převzata ze zdroje. [48]

```
void(* resetFunc) (void) = 0; // funkce pro reset
```

Obrázek 36 Funkce pro reset mikrokontroleru

8.5 Setup

Funkce **setup()** Obsahuje inicializaci vstupních a výstupních pinů včetně nastavení jejich prvotních hodnot, nastavení sériové komunikace a první měření sledovaných hodnot pro zapsání na displej. Nachází se zde také nastavení hodnoty hodin, které má nastavený čas první kompilace. Po první kompilaci byly hodiny již správně nastavené a kdyby tam tato část kódu zůstala aktivní, mělo by to za následek při každém spuštění přepsání času na prvotní hodnotu, která by byla v tu chvíli již mimo skutečný čas.

```
void setup()
{
  pinMode(pudaAnalog, INPUT);
  pinMode(vodaPin, INPUT);
  pinMode(svetloDigital, INPUT);
  pinMode(ventPin, OUTPUT);
  pinMode(alarmPin, OUTPUT);
  pinMode(cerpadloPin, OUTPUT);
  pinMode(vccPinPuda, OUTPUT);
  pinMode(vccPinSvetlo, OUTPUT);
  pinMode(svetloPin, OUTPUT);
  digitalWrite(ventPin, LOW);
  digitalWrite(alarmPin, LOW);
  digitalWrite(cerpadloPin, LOW);
  digitalWrite(vccPinPuda, LOW);
  digitalWrite(vccPinSvetlo, LOW);
  digitalWrite(svetloPin, LOW);
  serva.attach(servaPin);
  IN.begin();
  OUT.begin();
  rtc.begin();
  myNex.begin(9600);
  //rtc.setDateTime(__DATE__, __TIME__); //jednou provedene nastaveni hodin

  strHodiny = ctiCas();
  voda = testVoda();
  svetla = svetlo();
  vlhkostPuda = merPuda();
  teplotaIN = IN.readTemperature();
  teplotaOUT = OUT.readTemperature();
  zapis(strHodiny, svetla, vent, teplotaIN, teplotaOUT, voda, vlhkostPuda);
}
```

Obrázek 37 Funkce Setup

8.6 Loop

Funkce **loop()** obsahuje vlastní program, který běží v nekonečné smyčce. Jsou v ní použity předem definované uživatelské funkce. Je možné ji rozdělit na 4 časové intervaly, při kterých se provádí určité činnosti.

1. **Čtení a zápis**

   Tato akce se provádí pokud možno každou vteřinu. Je to zajištěné tím, že program si ukládá čas, kdy naposledy byla akce provedena, a pak tento čas porovnává s aktuálním časem. Jsou zde akce, které se musí provádět často, jako jsou čtení a zápis na displej, ale i kontrola vody a čtení reálného času z hodin kvůli zapisování správného času.

Program si přečte nastavená data z displeje pomocí funkce na čtení a potencionální změny nastavení se propíší do programu. Dále se zkontroluje stav vody pomocí funkce k tomu určené, přečte se čas z hodin a všechny tyto informace se zapíší na displej, u první iterace programu nenastává problém s hodnotami teplot a vlhkostí, z toho důvodu, že tyto informace jsou již uloženy z funkce setup. Nakonec se uloží aktuální hodnota času mikroprocesoru pro následné kontroly, zda už uběhl dostatečný čas pro opětovné spuštění této části kódu.

```
void loop()
{
  if (millis() - casCti > 1000)
  {
    cti();
    voda = testVoda();
    strHodiny = ctiCas();
    zapis(strHodiny, svetla, vent, teplotaIN, teplotaOUT, voda, vlhkostPuda);
    casCti = millis();
  }
```

Obrázek 38 Funkce v prvním časovém intervalu loop 1 s

2. Čtení teplot, ventilace a světel

Tyto akce se provádí jednou za 10 vteřin, jelikož není potřeba provádět je tak často, větší časový interval mezi měřením zvyšuje životnost čidel. Úkolem této části je změření a uložení hodnoty teploty uvnitř i venku, následná kontrola a případná správa ventilace a světel.

Nejdříve se změří teploty na čidlech a uloží se do proměnných. Poté budou použity ve funkci pro ventilaci a později budou i zapsány na displej (při opětovném průběhu části 1). Následně se provede funkce spravující světla, a nakonec se uloží čas mikrokontroleru pro kontrolu před opětovným spuštěním této části kódu.

```
if (millis() - casMer > 10000) //co 10s mereni 10000
{
  teplotaIN = IN.readTemperature();
  teplotaOUT = OUT.readTemperature();
  vent = ventilace(teplotaIN, teplotaOUT, teplotaMAX);
  svetla = svetlo();
  casMer = millis();
}
```

Obrázek 39 Funkce v druhém časovém intervalu cyklu loop 10 s

3. Čtení vlhkosti půdy, kontrola zalévání

V této části se změří vlhkost půdy a vyhodnotí se, zda je potřeba pustit zalévaní za předpokladu, že je v nádrži dostatek vody. Tato část probíhá jednou za 5 minut z důvodu, že není potřeba provádět ji častěji a zároveň se zvyšuje životnost půdního čidla na měření vlhkosti.

Jakmile proběhne měření vlhkosti půdy pomocí funkce merPuda, výsledek se uloží do proměnné. Poté bude použit při vyhodnocování, zda se má spustit zalévání a zapsán na displej (to až při opětovném průběhu první části). Při samotném vyhodnocování, jestli se má spustit zalévací čerpadlo, se použije porovnání, zda je ideální vlhkost půdy menší než skutečná a jestli je v nádržce na vodu dostatek vody (tuto informaci máme již z části 1). Vyhodnotí-li se, že je potřeba pustit zalévání, zapne se čerpadlo na 5 vteřin. V případě, že nebude i potom vlhkost půdy dostatečná, tato akce se bude opakovat každých 5 minut, dokud tomu tak nebude. Nakonec se uloží aktuální čas, pro kontrolu opětovného spuštění této části kódu.

```
if (millis() - casPuda > 300000) //co 5 minut mereni pudy 300000
{
  vlhkostPuda=merPuda();
  if(vlhkostPuda < vlhkostPMIN && voda == 0)
  {
    cerpadlo(5000);
  }
  casPuda = millis();
}
```

Obrázek 40 Funkce ve třetím časovém intervalu cyklu loop 5 m

4. Reset

Slouží jen pro celkový reset mikrokontroleru z důvodu rizika přetečení hodin, jelikož každá paměť je omezená a do proměnné ukládající si hodnotu o čase se nevejde číslo vetší než 2^32-1, což je 4294967295 ms a to se rovná cca 49 dnům. Proto je reset nastaven na 4200000000 ms (přibližně 48.61 dne).

```
  if (millis() > 4200000000) //reset po 48.6 dnech 4200000000 kvuli preteceni hodin po 49 dnech
  {
    resetFunc(); //call reset
  }
}
```

Obrázek 41 Funkce ve čtvrtém časovém intervalu cyklu loop 48.61 dne

8.7 Displej Nextion

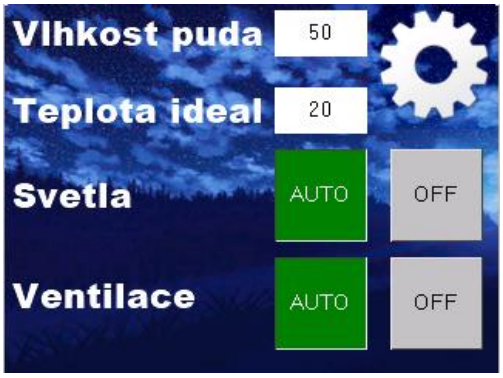
8.7.1 Rozložení

Displej je rozdělen na 2 stránky (page0 a page1). Úkolem nulté stránky je zobrazovat informace pro uživatele, zatímco na první se provádí nastavování samotného systému.

Na nulté stránce se nacházejí informace o času, stavu světel a ventilace, aktuálních teplotách uvnitř i mimo skleník, stavu nádrže s vodou a vlhkosti půdy. V horním pravém rohu je umístěné tlačítko (button) pro přechod na první stránku s nastavením. Bystřejší oko si muže všimnout bílých teček v krajích displeje. To jsou numerická pole, ve kterých jsou uloženy nastavené hodnoty. Jsou zde, protože byl při programováni problém s nastavováním globálních proměnných a toto bylo elegantní řešení, jak tento problém obejít.

Obrázek 42 Ukázka rozložení page0 na displeji

Na první stránce je možné vykonávat nastavování hodnot, tedy nastavení ideální vlhkosti půdy a ideální teploty uvnitř skleníku. Dále je zde možné přes tlačítka ovládat režimy světel a ventilace. Lze vybírat mezi automatickým režimem řízeným mikrokontrolerem, režimem ON, čímž uživatel zvolí, zda chce, aby světla/ventilace běžela nehledě na rozhodování mikrokontroleru, anebo režim OFF, kdy jsou systémy světel/ventilace vypnuté bez ohledu na rozhodování mikrokontroleru.



Obrázek 43 Ukázka rozložení page1 na displeji

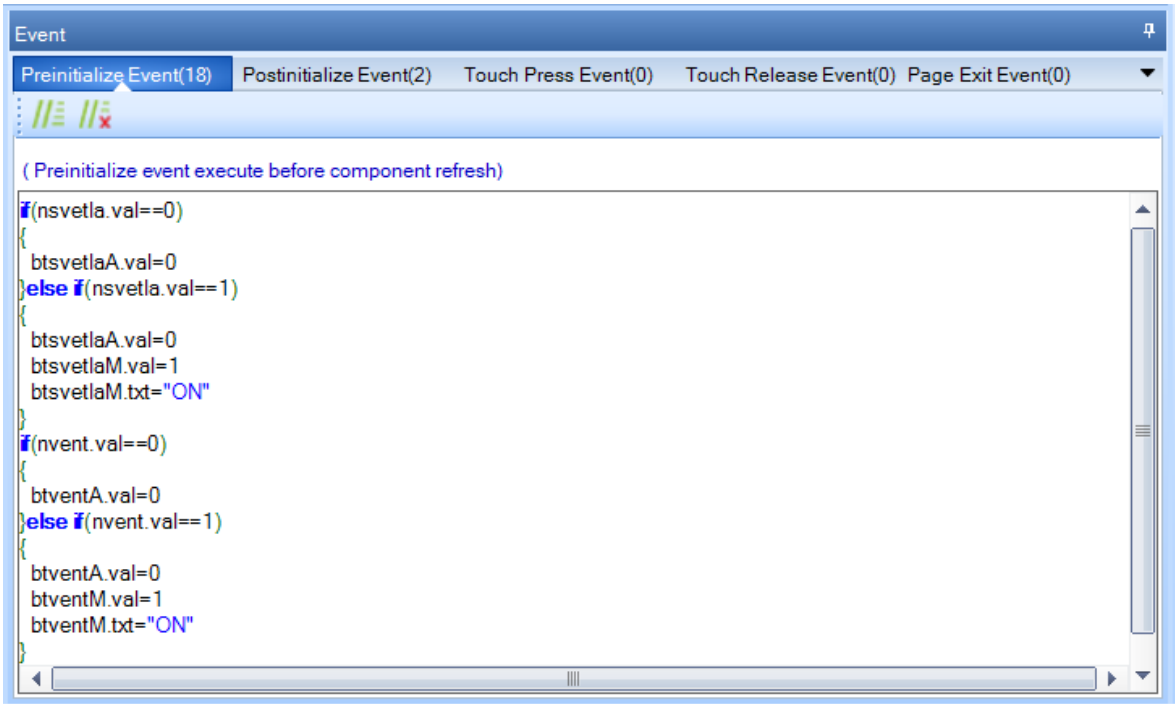
8.7.2 Stránky

Stránkám bylo zapotřebí udat základní funkční prvky, jako jsou doba, po které se přepnou do režimu spánku, a způsob, jakým je lze z režimu spánku probrat, nebo jaká mají provést nastavení přepneme-li se na ně.

Na stránce nula bylo zapotřebí jen nastavit režim spánku, který proběhne po 30 vteřinách nečinnosti uživatele na displeji pomocí příkazu „thsp“, jehož parametrem je počet vteřin od poslední akce, v tomto případě 30. Dále je nastaveno, aby se probral z režimu spánku po dotyku pomocí příkazu „thup“ s parametrem 1.

Obrázek 44 Funkce stránky page0. Foto autor

Na stránce jedna toho ale bylo zapotřebí nastavit více. Režim spánku a probuzení je zde naprosto stejný, bylo však potřeba nastavit přepínání tlačítek při přechodu mezi stránkami, jelikož kdykoliv by se přepnulo ze stránky nula na stránku jedna, byly by pozice tlačítek v defaultní pozici, což by bylo po jakékoliv provedené změně nastavení nepřijatelné. To zabezpečuje kód stránky, který proběhne vždy při načtení stránky. Princip je následovný, defaultní pozice je s číslem 2 (automatický mód, tlačítko AUTO svítí a tlačítko pro manuální ovládání bude v zhasnuté pozici s textem OFF), proto nemusí být řešena, vrátí-li se ale pozice s číslem 0, musí být tlačítka správně nastavena (pozice 0 znamená, že je funkce ručně vypnuta, tudíž AUTO zhasnuto a u manuálu je zobrazeno OFF). Vrátí-li se pozice s číslem 1, znamená to, že má být činnost spuštěna bez ohledu na okolnosti, tudíž AUTO zhasnuté a tlačítko pro manuální ovládání rozsvíceno s nápisem ON. Tento princip funguje jak pro ventilaci, tak světla.



Obrázek 45 Funkce stránky page1. Foto autor

8.7.3 Tlačítka

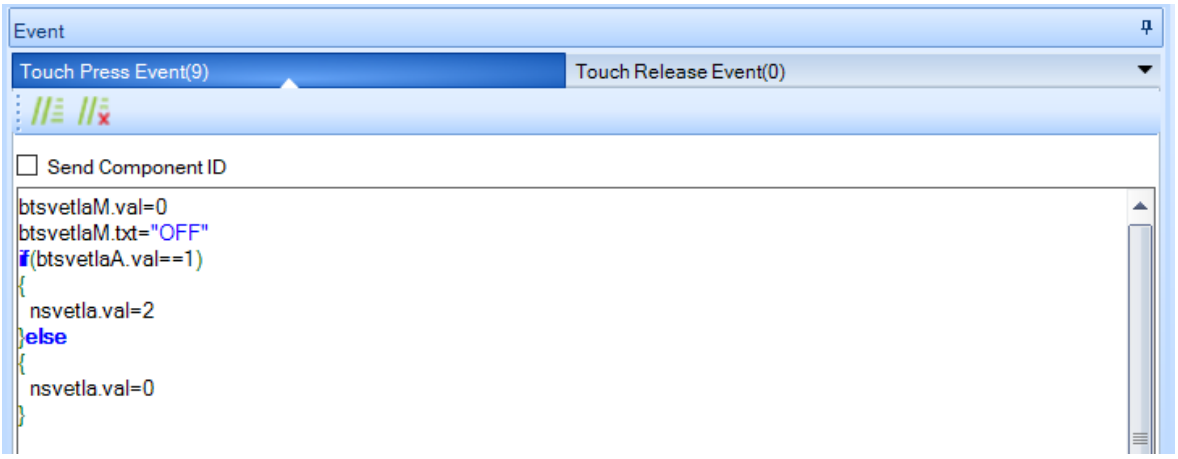
Sama tlačítka musí mít svou funkční část. Každý stisk tlačítka provádí určité změny a ty jsou zapsány určitými příkazy.

Obrázek 46 Ukázka funkčních tlačítek a polí

Tlačítko btsvtelaA (button svetla Automat)

Tlačítko sloužící k zapínaní/vypínání automatického režimu světel. Stiskne-li se, vždy nastaví tlačítko btsvetlaM na 0, tedy zhasnuté s nápisem OFF. Bylo-li předtím zhasnuté a stiskem změnilo svou hodnotu (btsvetlaA.val) na 1, znamená to, že byl aktivován automatický režim a do proměnné reprezentující aktuální mód světel se zapíše 2. Svítilo-li,

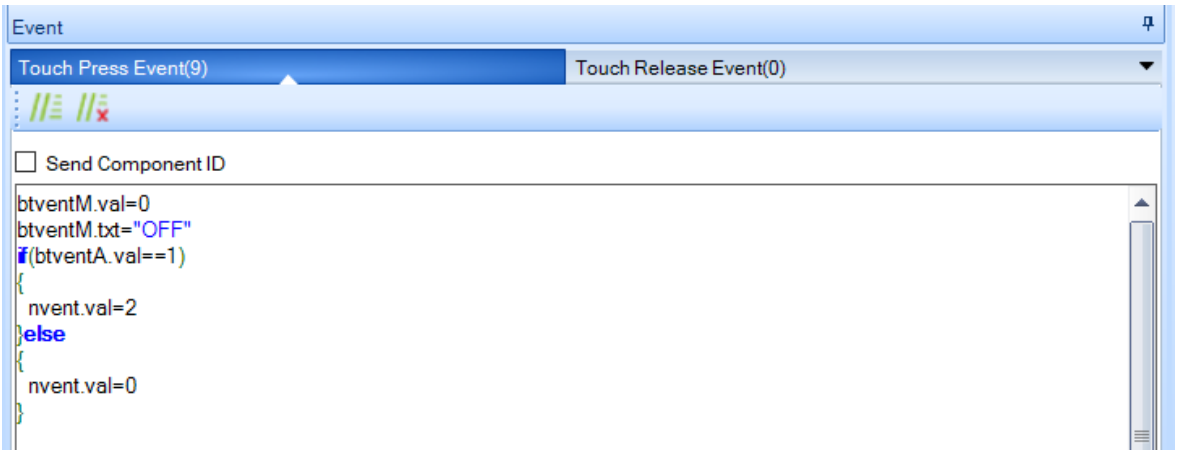
po stisku změnilo svou hodnotu (btsvetlaA.val) na 0 a to znamená, že chceme vypnout automatický režim. Tlačítko po stisku automaticky zhasne a do proměnné reprezentující aktuální mód světel se zapíše 0.



Obrázek 47 Funkce tlačítka automatických světel

Tlačítko btventA (button ventilace Automat)

Tlačítko funguje principiálně naprosto stejně jako tlačítko btsvetlaA, jen s jinými názvy a místo světel ovládá ventilaci.

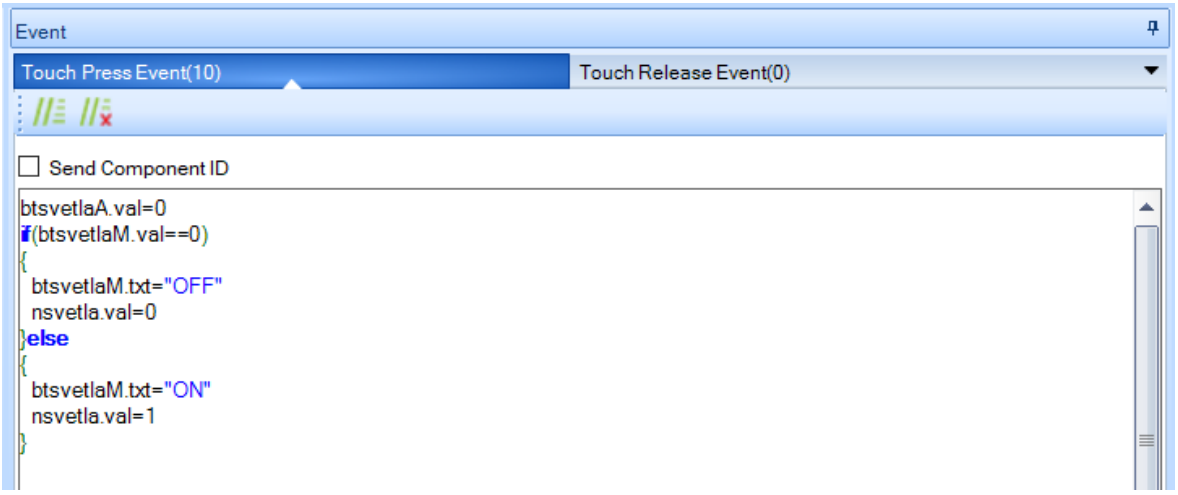


Obrázek 48 Funkce tlačítka automatické ventilace

Tlačítko btsvtelaM (button svetla Manual)

Tlačítko sloužící k nastavování manuálního ovládaní světel. Stiskne-li se, automatický režim se vypne (hodnota na btsvetlaA.val se přepíše na 0, což má za následek, že tlačítko zhasne, dále se vyhodnotí podmínka, v jakém je teď tlačítko stavu, a podle toho buď zhasne

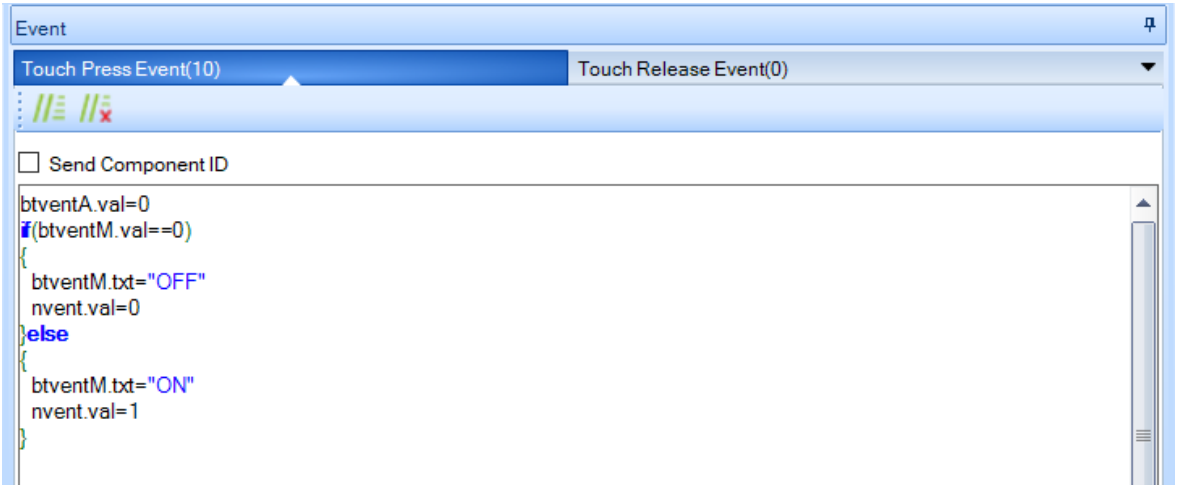
a přepíše na OFF, nebo rozsvítí a přepíše na ON. Následně se do proměnné reprezentující aktuální mód světel zapíše 0 (OFF) nebo 1 (ON) podle situace.



Obrázek 49 Funkce tlačítka manuálního ovládaní světel. Foto autor

Tlačítko btventM (button ventilace Manual)

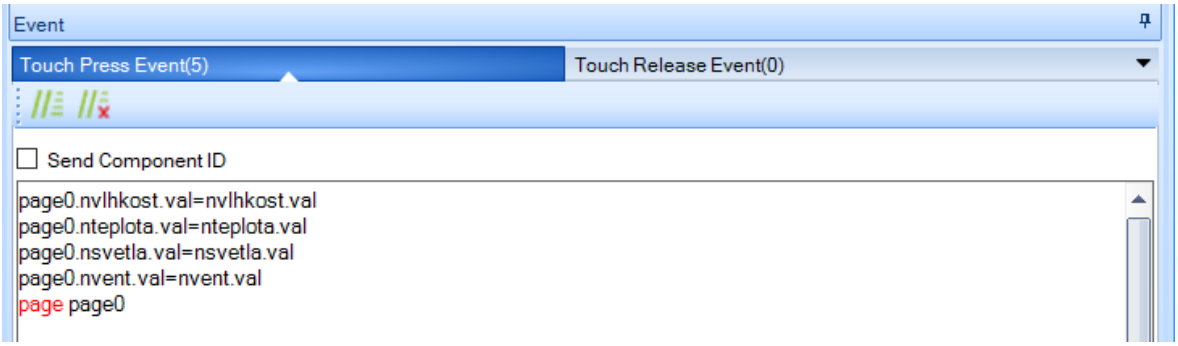
Tlačítko funguje principiálně naprosto stejně jako tlačítko btsvetlaM, jen s jinými názvy a místo světel ovládá ventilaci.



Obrázek 50 Funkce tlčítka manuálního ovládání ventilace

Tlačítko b0

Toto tlačítko slouží k předávání informací mezi stránkami, resp. informace o nastavené ideální teplotě, vlhkosti půdy, režimů světel a ventilace se odešlou a zapíší na druhou stránku. Nakonec se stránka přepne. Toto tlačítko se nachází na obou stránkách. Jediný rozdíl je tedy jen v použitých názvech stránky (page0, page1).



Obrázek 51 Funkce tlačítka na přepínaní stránek

.

8.8 Sestavení, nastavení a ovládání

Po sestavení je většinová část ukryta v plastovém boxu vytisknutém na 3D tiskárně. Přesněji se v něm nachází displej, deska Arduino, nepájivé spojové pole, sensor pro měření tepoty venku, RTC hodiny a signalizační bzučák spolu s LED diodou indikující nedostatek vody v nádržce. Do boxu je přivedené napájení z elektrické sítě a systém se automaticky zapíná hned po zapojení do zásuvky. Z boxu vedou vodiče k sensorům, ventilaci a osvětlení. Sensor na měření teploty uvnitř skleníku je umístěn na stěně skleníku, stejně tak sensor osvětlení. Půdní sensor vlhkosti je zapíchnut do zeminy. Čerpadlo je přilepeno na dno nádrže na vodu a vede z něj PVC hadička, která je vyvedena u rostliny, zpět do skleníku. Plovákový sensor je umístěn u dna nádrže, pár centimetrů nad sáním čerpadla, aby se zajistila včasná signalizace nedostatku vody a nedošlo k poškození čerpadla během na prázdno. Světla jsou umístěna ve vrchní části skleníku, aby pokryla co největší plochu. Ventilace se skládá ze dvou ventilátorů a dvou serv. Serva jsou použita pro otvírání a zavíraní průduchů před ventilátory. Ventilátory jsou použity k podpoření proudění vzduchu skrze skleník. Veškeré vodiče vedoucí z boxu dále do skleníku jsou izolovány, aby se zamezilo riziku zkratu.

Po zapnutí je prvotní nastavení nastaveno na plně automatické s ideální vlhkostí půdy 50 % a teplotou vzduchu 20 °C. Veškeré další ovládání probíhá z displeje ze stránky 2, na kterou se uživatel dostane stisknutím ozubeného kolečka v pravém horním rohu displeje.

9 Závěr

V této práci jsem se věnoval problematice skleníků a jejich automatizace, za účelem snížení rizika lidského faktoru na růst rostlin. Čtenář byl seznámen s principy automatizace, skleníků a platformou Arduino, včetně porovnání i s jinými platformami. Dále byl čtenář seznámen s komponenty, které byly použity při realizaci projektu.

Při realizaci jsem narazil i na problémy, kdy bylo zapotřebí ošetřit vstupy od uživatele, abych zamezil případnému poškození jednotlivých komponentů. Dále jsem narazil na problém s převáděním analogové hodnoty na reálné číslo ukazující vlhkost půdy, který jsem vyřešil nemapováním analogové hodnoty po provedené kalibraci čidla. Při komunikaci mezi deskou a displejem docházelo k chybnému čtení hodnot z objektů na displeji. Tento problém jsem z velké části programově eliminoval. Bohužel se ale nepovedlo ho zcela vyřešit. Nepodařilo se mi určit, kde přesně ještě může docházet k tomuto problému. Napadají mě dvě možnosti. První je, že může být problém s napájení displeje přímo z desky Arduino, bohužel na externí napájeni displeje jsem již neměl zdroje. Druhá možnost je, že může dojít k problému, když uživatel uloží data do objektu na displeji v moment, kdy je prováděno čtení. To se mi ale při testováni nepovedlo ověřit a z logického hlediska by v tento moment mělo zafungovat programové ošetření. Další komplikace způsobilo nízké napětí, které mě donutilo změnit moji myšlenku ze sériové zapojeného osvětlení led diodami do série na zapojení paralelní. Diody zapojené v sérii měly nízkou svítivost, a jiné nesvítily vůbec. Problém byl i s PWM piny, které neposkytovaly dostatečné napětí pro provoz čerpadla a dalších komponentů, tento problém jsem vyřešil použitím NPN tranzistorů, které posloužily jako spínače mezi komponenty a 5V piny desky. Po vyřešení těchto problému je systém plně funkční a plní zadané požadavky. Jelikož jde k desce Arduino připojovat další hardware bez větších problémů, dal by se projekt snadno rozšiřovat přidáváním dalších komponentů a menší programovou úpravou.

10 Zdroje

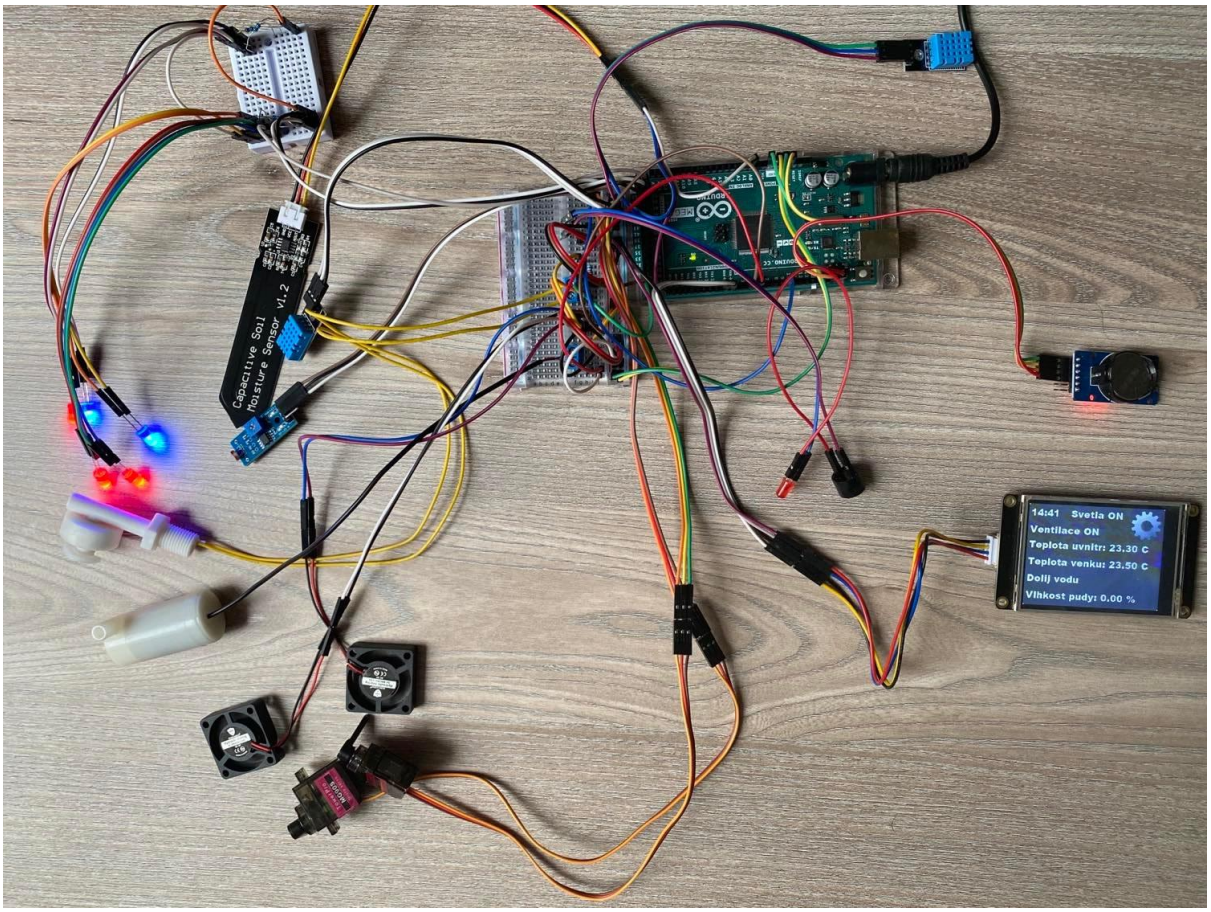
[1] **xzboril7.** ate.pdf. *https://akela.mendelu.cz/.* [Online] [Citace: 6. 2 2022.] Dostupné z: https://akela.mendelu.cz/~xzboril7/ate.pdf.
[2] **Žáček, Michal.** Historicky vyvoj automatizace poznejte 12 zasadnich dat. *factoryautomation.cz.* [Online] 17. 3 2015. [Citace: 6. 3 2022.] Dostupné z: https://factoryautomation.cz/historicky-vyvoj-automatizace-poznejte-12-zasadnich-dat/.
[3] **plusSystem.** 5 důvodů Proč Automatizovat. *plussystem.cz.* [Online] 24. 2 2021. [Citace: 7. 3 2022.] Dostupné z: https://www.plussystem.cz/5-duvodu-proc-automatizovat/.
[4] **Pastyříková, Veronika.** *Založení, historie a současnost skleníkového areálu ve Smetanových sadech v Olomouci*. [Online] Olomouc 2014. Dostupné z: https://theses.cz/id/c3vsa4/Pastyrikova_Veronika_-_Zalozeni_historie_a_soucasnost_skl.pdf. Bakalářská práce. Univerzita Palackého v Olomouci. Pedagogická fakulta. Vedoucí práce: Ing. Pavlína Škardová.
[5] **Paprsek, Adam.** *AUTOMATICKÝ SKLENÍK PRO PĚSTOVÁNÍ ROSTLIN*. [Online] Brno 2015. [Citace: 7. 3 2022.] Dostupné z: https://dspace.vutbr.cz/bitstream/handle/11012/41531/final-thesis.pdf?sequence=10&isAllowed=y. Bakalářská práce. Vysoké učení technické v Brně. Fakulta komunikačních technologií ústav mikroelektroniky. Vedoucí práce: Ing. Ladislav Macháň.
[6] **Pavel Chlouba, Hana Šoberová.** Skleniky byly jen pro bohate. Zmenila to prumyslova revoluce a zruseni dane ze skla. *budejovice.rozhlas.cz.* [Online] © 1997-2022 Český rozhlas, 23. 2 2018. [Citace: 12. 3 2022.] Dostupné z: https://budejovice.rozhlas.cz/skleniky-byly-jen-pro-bohate-zmenila-prumyslova-revoluce-a-zruseni-dane-ze-skla-7035800.
[7] **Ing.Martin Koudela, Ph.D.** Ekologická produkce zeleniny. *Ekologická produkce zeleniny.* [Online] [Citace: 14. 3 2022.] Dostupné z: http://kz.agrobiologie.cz/ekozem/?m=obecna&p=obecna_faktory.
[8] **Doskočil, Jakub.** Návrh a konstrukce vzdáleného experimentu - meteorologická stanice. [Online] Olomouc 2015. [Citace: 7. 3 2022.] Dostupné z: https://theses.cz/id/wajj2j/15004641 Bakalářská práce. Univerzita Palackého v Olomouci. Pedagogická fakulta. Vedoucí práce PhDr. PaedDr. Jiří Dostál, Ph.D.
[9] **K., Webster.** 1. díl o Arduinu – Historie. *PHGame.cz.* [Online] PHGame, 5. 3 2015. [Citace: 7. 3 2022.] Dostupné z: https://phgame.cz/PHGame_serialy/serialy/zaciname-s-arduinem/1-dil-o-arduinu-historie/.
[10] **Ježková, Adéla.** *Využití Raspberry Pi v domácnosti*. [Online] Praha 2018. Dostupné z: https://insis.vse.cz/zp/66954 Bakalářská práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce: Ing. Ladislav Luc.
[11] **PICAXE.** What Is PICAXE? - What is PICAXE. *PICAXE.* [Online] [Citace: 8. 3 2022.] Dostupné z: https://picaxe.com/what-is-picaxe/.
[12] **Sumida Crossing.** Feather M0 Microprocessor. *Sumida Crossing.* [Online] [Citace: 7. 3 2022.] Dostupné z: http://www.sumidacrossing.org/LayoutElectricity/microcontrollers/M0Feather/.
[13] **Arduino.cc.** Arduino Integrated Development Environment (IDE) v1. *docs.arduino.cc.* [Online] © 2021 Arduino, 24. 2 2022. [Citace: 8. 3 2022.] Dostupné z: https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics.
[14] **Wiring.** About \ Wiring. *Wiring cover.* [Online] [Citace: 7. 3 2022.] Dostupné z: http://wiring.org.co/about.html. ISSN 2011-8376.

[15] **Internet of Things Project.** Wiring Programming Language - Internet of Things Project. *Internet of Things Project.* [Online] 2019. [Citace: 7. 3 2022.] https://docs.idew.org/internet-of-things-project/references-for-wiring-and-coding/wiring-programming-language.
[16] **Digi-Key Electronics.** Multi-tasking the Arduino - Part 1. *Digi.* [Online] Digi-Key Electronics. [Citace: 7. 3 2022.] Dostupné z: https://www.digikey.cz/en/maker/projects/multi-tasking-the-arduino-part-1/b23d9e65c4d342389d20cbd542c46a28.
[17] **hobbyrobot.cz.** [Online] [Citace: 8. 3 2022.] Dostupné z: http://www.hobbyrobot.cz/wp-content/uploads/ArduinoPriruckaProgramatora.pdf.
[18] **Voda, Zbyšek.** Uživatelsky definované funkce. *Bastlírna HWKITCHEN.* [Online] 12. 5 2014. [Citace: 8. 3 2022.] Dostupné z: https://bastlirna.hwkitchen.cz/uzivatelsky-definovane-funkce-2/.
[19] **Arduino.cc.** Arduino Uno Rev3. *Arduino Official Store.* [Online] © 2021 Arduino. [Citace: 7. 3 2022.] Dostupné z: http://store.arduino.cc/products/arduino-uno-rev3.
[20] **Arduino.cc**. [foto].Pinout-UNOrev3_latest.png. *content.arduino.cc.* [Online] © 2021 Arduino. [Citace: 16. 2 2022.] Dostupné z: https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf.
[21] **Arduino.cc**. Arduino Nano. *Arduino Official Store.* [Online] © 2021 Arduino. [Citace: 7. 3 2022.] Dostupné z: https://store.arduino.cc/products/arduino-nano.
[22] **Arduino.cc**. [foto].Pinout-NANO_latest.png. *content.arduino.cc.* [Online] © 2021 Arduino. [Citace: 16. 2 2022.] Dostupné z: https://content.arduino.cc/assets/Pinout-NANO_latest.png.
[23] **Arduino.cc** Arduino Leonardo with Headers. *Arduino Official Store.* [Online] © 2021 Arduino. [Citace: 8. 3 2022.] Dostupné z: http://store.arduino.cc/products/arduino-leonardo-with-headers.
[24] **Arduino.cc.** [foto].Pinout-Leonardo_latest.png. *content.arduino.cc.* [Online] © 2021 Arduino. [Citace: 6. 2 2022.] Dostupné z: https://content.arduino.cc/assets/Pinout-Leonardo_latest.png.
[25] **Arduino.cc**. Arduino Mega 2560 Rev3. *Arduino Official Store.* [Online] © 2021 Arduino. [Citace: 8. 3 2022.] Dostupné z: http://store.arduino.cc/products/arduino-mega-2560-rev3.
[26] **Arduino.cc**. [foto].Pinout-Mega2560rev3_latest.png. *content.arduino.cc.* [Online] © 2021 Arduino. [Citace: 6. 2 2022.] Dostupé z: https://content.arduino.cc/assets/Pinout-Mega2560rev3_latest.png.
[27] **laskakit.cz.** Ponorné mini čerpadlo ultra-tiché DC 3-6V 120 L/H. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz, 26. 3 2021. [Citace: 6. 1 2022.] Dostupné z: https://www.laskakit.cz/ponorne-mini-cerpadlo-ultra-tiche-dc-3-6v-120-l-h/?variantId=1109.
[28] **laskakit.cz**. [foto].965-2_ponorne-mini-cerpadlo-ultra-tiche-dc-3-6v-120-l-h.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/965-2_ponorne-mini-cerpadlo-ultra-tiche-dc-3-6v-120-l-h.jpg?61d95cfc.
[29] **laskakit.cz**. Plovákový senzor vodní hladiny, vodorovný. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz, 11. 2 2022. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/plovakovy-senzor-vodni-hladiny--vodorovny/.
[30] **laskakit.cz**. [foto].5674-2_5674-2-plovakovy-senzor-vodni-hladiny-vodorovny.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z:

https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/5674-2_5674-2-plovakovy-senzor-vodni-hladiny-vodorovny.jpg?6137b46c.
[31] **laskakit.cz**. Mini servo MG90S s kovovými převody. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/mini-servo-mg90s-s-kovovymi-prevody/.
[32] **laskakit.cz**. [foto].341_mini-servo-mg90s-s-kovovymi-prevody.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/341_mini-servo-mg90s-s-kovovymi-prevody.jpg?61d95cec.
[33] **laskakit.cz**. ASAIR senzor teploty a vlhkosti vzduchu DHT11, modul. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz, 21. 1 2022. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/arduino-senzor-teploty-a-vlhkosti-vzduchu-dht11--modul/.
[34] **laskakit.cz**. [foto].563_arduino-senzor-teploty-a-vlhkosti-vzduchu-dht11--modul.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/563_arduino-senzor-teploty-a-vlhkosti-vzduchu-dht11--modul.jpg?61d95caa.
[35] **laskakit.cz**. Kapacitní čidlo pro měření vlhkosti půdy. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 27. 2 2022.] Dostupné z: https://www.laskakit.cz/kapacitni-cidlo-pro-mereni-vlhkosti-pudy/.
[36] **laskakit.cz**. [foto].1931_kapacitni-cidlo-pro-mereni-vlhkosti-pudy.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/1931_kapacitni-cidlo-pro-mereni-vlhkosti-pudy.jpg?61d95cd4.
[37] **laskakit.cz**. RTC Hodiny reálného času DS3231 AT24C32. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz, 22. 2 2022. [Citace: 27. 2 2022.] Dostpný z: https://www.laskakit.cz/arduino-rtc-hodiny-realneho-casu-ds3231-at24c32/.
[38] **laskakit.cz**. [foto].272_arduino-rtc-hodiny-realneho-casu-ds3231-at24c32.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/272_arduino-rtc-hodiny-realneho-casu-ds3231-at24c32.jpg?61d95d25.
[39] **laskakit.cz**. Světelný senzor, 4 pin modul. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/arduino-svetelny-senzor--4-pin-modul/.
[40] **laskakit.cz**. [foto].1886_arduino-svetelny-senzor--4-pin-modul.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/1886_arduino-svetelny-senzor--4-pin-modul.jpg?61d95d37.
[41] **laskakit.cz**. Nextion orig. Enhanced NX3224K028 2.8" 320 x 240 TFT displej. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/nextion-orig--enhanced-nx3224k028-2-8--320-x-240-tft-displej/.
[42] **laskakit.cz**. [foto].7038-1_7038-1-nextion-orig-enhanced-nx3224k028-2-8-320-x-240-tft-displej.jpg. *cdn.myshoptet.com.* [Online] Copyright 2022 laskakit.cz. [Citace: 6. 2 2022.] Dostupné z: https://cdn.myshoptet.com/usr/www.laskakit.cz/user/shop/big/7038-1_7038-1-nextion-orig-enhanced-nx3224k028-2-8-320-x-240-tft-displej.jpg?6141b2a0.
[43] **laskakit.cz**. LED dioda 5mm. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz, 25. 1 2022. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/led-dioda-5mm/?variantId=1127.
[44] **tme.eu.** Rezistory. *tme.eu.* [Online] Copyright © 2022 TME. [Citace: 16. 2 2022.] Dostupné z: https://www.tme.eu/cz/katalog/rezistory_100299/.

[45] **arduinoslovakia.eu.** Pro začátečníky: NPN tranzistor jako spínač. *arduinoslovakia.eu.* [Online] Copyright © 2013-2020 Róbert Ulbricht. [Citace: 16. 2 2022.] Dostupné z: https://www.arduinoslovakia.eu/blog/2019/7/pre-zaciatocnikov--npn-tranzistor-ako-spinac?lang=cs.
[46] **laskakit.cz.** Aktivní bzučák 5V. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/aktivni-bzucak-5v/.
[47] **laskakit.cz**. Nepájivé kontaktní pole 400 pinů, Bílé. *laskakit.cz.* [Online] Copyright 2022 laskakit.cz. [Citace: 16. 2 2022.] Dostupné z: https://www.laskakit.cz/nepajive-kontaktni-pole-400-pinu--bile/.
[48] **ondraN.** Kdy reaguje reset PIN - forum.hwkitchen.cz. *forum.hwkitchen.cz.* [Online] 24. 9 2020. [Citace: 6. 3 2022.] Dostupné z: https://forum.hwkitchen.cz/viewtopic.php?t=2460.

11 Přílohy



Příloha 1 Zapojený projekt před montáží