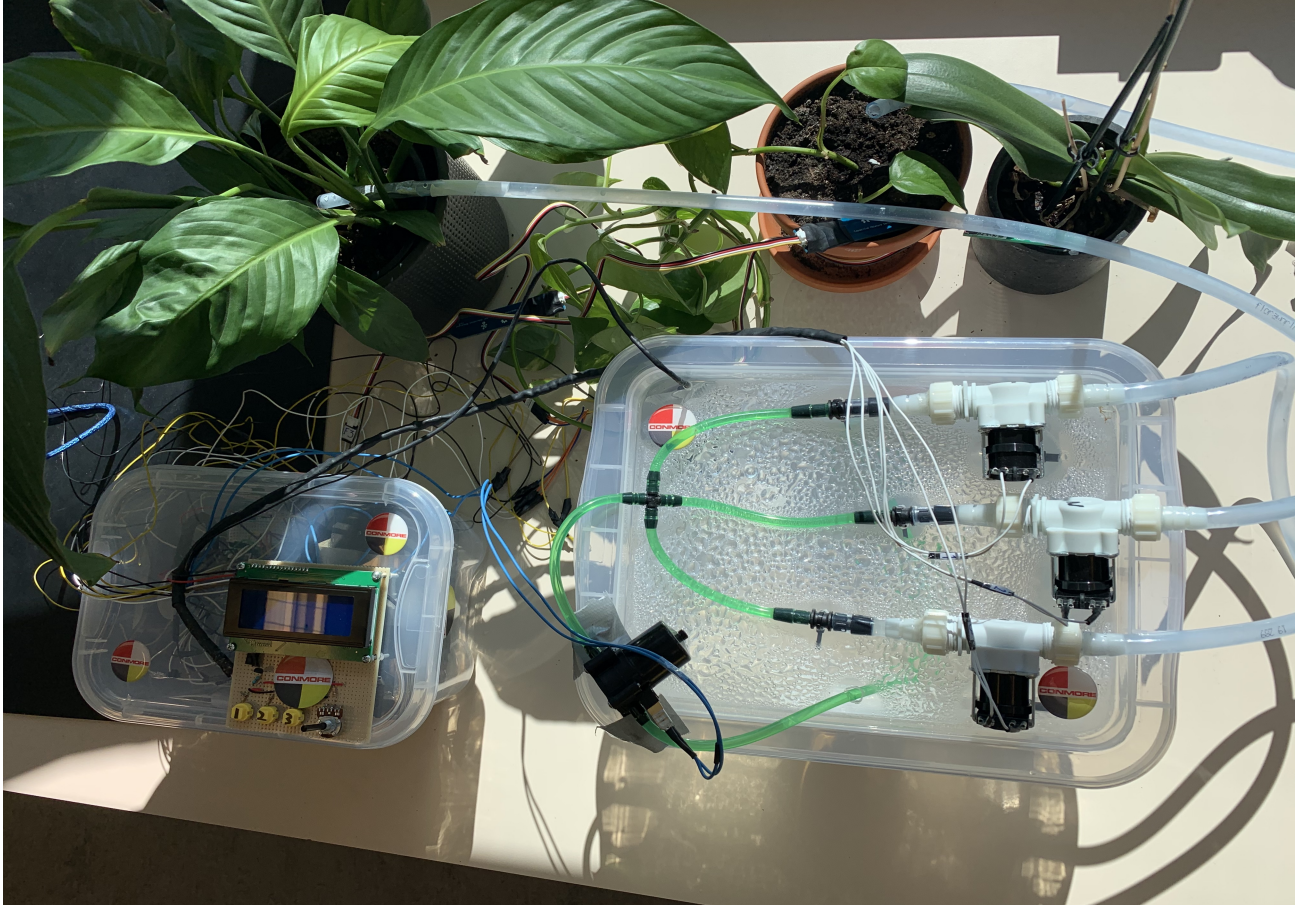




CHALMERS



Automatiskt bevattningssystem för krukväxter med Arduino® utvecklingskort

Ett examensarbete inom högskoleingenjörsprogramet Mekatronik

Automatic watering system for house plants with an Arduino®

A bachelor thesis for the Mechatronics programme

Emma Bergquist
Victoria Bigert

EXAMENSARBETE INOM MEKATRONIK 2020

**Automatiskt bevattningssystem
för krukväxter med Arduino[®] utvecklingskort**

**Automatic watering system
for house plants with an Arduino[®]**

EMMA BERGQUIST
VICTORIA BIGERT



CHALMERS

Institutionen för Elektroteknik
Avdelningen för system- och reglerteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2020

**Automatiskt bevattningssystem för krukväxter
med Arduino[®] utvecklingskort**
EMMA BERGQUIST
VICTORIA BIGERT

Handledare: Fredrik Hansson, Conmore
Examinator och handledare: Bertil Thomas, Institutionen för Elektroteknik

Institutionen för Elektroteknik
Avdelningen för system- och reglerteknik
Chalmers tekniska högskola
SE-412 96 Göteborg, Sverige
Telefon till växeln: +46 31 772 1000

Omslagsbild: Bild på färdiga bevattningssystemet, Göteborg 22-4-20

Sammanfattning

Det kan vara svårt att ge rätt mängd vatten som växter behöver. Detta projekt handlar om hur ett system kan styra bevattningen automatiskt med enbart några få direktiv från en människa. Projektet har utgått från en Arduino[®], en pump och tre magnetventiler. Dessa fem utgångspunkter tillsammans med olika elektriska komponenter bildar en prototyp som kan styra bevattningen. Arduinon[®] har programmerats i utvecklingsmiljön Arduino[®] IDE. Programmet utgår från att hålla fukthalten i jorden lik den angivna fukthalten som växten ska ha. Prototypen genomgick en testperiod där funktioner och hårdvara testades för att bland annat ta fram de tidskonstanter som lämpade sig bäst för systemet.

Resultatet blev ett fungerande system som kan hålla en jämn fukthalt i jorden. Anordningen kan dessutom få jorden att torka ut innan det vattnar på nytt. Det finns också valet om växten ska få mycket eller lite vatten.

Keywords: Automation, växter, prototyp, Arduino[®], magnetventiler, givare

Abstract

Watering plants with just the right amount of water is hard. This project shows how a system can take care of the watering of house plants automatically with only a couple of directives from a human. Everything started with an Arduino[®], a pump and three magnetic solenoid valves. These five starting points with some electrical components make up a prototype that can steer the automatic watering of house plants. The Arduino[®] was programmed in the environment Arduino[®] IDE. The programme proceeds from keeping the moisture percent in the soil the same as the set point for the moisture the plant is supposed to have. The prototype went through a testphase were functions and hardware was tested, for example to get the needed time constants for the system.

The results were a prototype that could keep an even moisture percent in the soil. The system also has the ability to make the soil dry out before it gets water again. Flower pots come in different sizes and therefore need more or less water. Because of that a function to let the system give different amount of water when watering.

Keywords: Automatic, Arduino[®], magnetic solenoid, moisture, sensor

Förkortningar

AD - Analog to Digital (Analog till digital omvandling)

BJT - Bipolar Junction Transistor (Bipolärtransistor)

FET - Field Effect Transistor (Fälteffekttransistor)

IDE - Integrated development environment (Integrerad Utvecklingsmiljö)

JFET - Junction Field Effect Transistor

LCD - Liquid Crystal Display

MCU - Microcontrolling Unit (Mikrokontroller)

MOSFET - Metal Oxide Semiconductor Field Effect Transistor

SCL - Serial Clock (Seriedata)

SDA - Serial Data (Serieklocka)

Förord

Examensarbetet är för Mekanik högskoleingenjör, Chalmers Tekniska Högskola. Projektet har genomförts inom institutionen för elektroteknik. Det har utförts åt företaget Conmore Ingenjörbyrå AB, där idén uppkom genom en presentation med examensarbete-idéer som Conmore hade färdig. Projektet pågick i 12 veckor och totalt med rapportskrivning och redovisning cirka 16 veckor.

Vi vill tacka Conmore och speciellt Fredrik Hansson för att vi fick göra projektet åt företaget, för ett gott samarbete och för hjälp och feedback på olika delar av projektet. Vi vill tacka Göran Stigler för hjälp med inköp av komponenter och vägledning i början av projektet. Vi tackar även Göran Hult för att vi fick komponenter och mycket tips och hjälp med koppling av olika delar. Vi vill slutligen tacka Bertil Thomas som varit vår handledare och examinator inom examensarbetet.

Emma Bergquist och Victoria Bigert, Göteborg 2020

Innehåll

1	Inledning	1
1.1	Bakgrund	1
1.2	Syfte	1
1.3	Avgränsningar	1
1.4	Precisering av mål	2
1.5	Arbetsmetod	4
2	Teori	5
2.1	Arduino®	5
2.2	Beskrivning av elektronikkomponenter	6
2.2.1	Magnetventiler	6
2.2.2	Universal spolarpump	7
2.2.3	LCD	8
2.2.4	Strömställare / Strömbrytare	8
2.2.5	Fuktighetsgivare	8
2.2.6	Temperaturgivare	9
2.2.7	Vridpotentiometer	9
2.2.8	Transistor	10
2.3	Beskrivning av signalbehandling: AD-omvandling	11

2.4	De valda växterna	11
3	Genomförande	12
3.1	Förarbete	12
3.2	Tidsplan	12
3.3	Fuktighetsgivare	12
3.4	Temperaturgivare	13
3.5	LCD	13
3.6	Pump	14
3.7	Nivågivare	14
3.8	Magnetventiler	15
3.9	Kombination av pump och magnetventiler	16
3.10	Drivning av prototypen	16
3.11	Elektronikkomponenter	16
3.11.1	Potentiometer	17
3.11.2	Strömbrytare / knappar	17
3.11.3	MOSFET	18
3.12	Koppling av el-komponenter	18
3.13	Programbeskrivning	20
3.14	Regulatorer	22

3.14.1	Vanlig regulator	22
3.14.2	Torkningsregulator	23
3.14.3	Hög temperatur	23
3.14.4	Mycket eller lite vatten	24
3.15	Test av prototyp	25
3.16	Säkerhetsfunktion	30
4	Resultat	31
4.1	Felkällor	31
5	Slutsats och Diskussion	32
5.1	Förbättringar	32
5.2	Hållbarhet ur miljöperspektiv	32
5.3	Slutsats	33

1 Inledning

Bevattningen i växthus för grödor och plantor har traditionellt gjorts för hand med vattenkannor, vattenslangar och andra medel. Med samhällets intresse för automatisering och elektrifiering finns det idag system som kan vattna växterna helt automatiskt. Genom detta har det hjälpt att effektivisera den egna skötseln i växthus men också att spara på vatten och arbetskraft som annars behövdes innan.

1.1 Bakgrund

Krukväxter är en väldigt populär inredningsdetalj i de flesta hushåll. Ett återkommande problem är att ta hand om plantorna när hushållet står tomt över en längre tid, som vid en semester. Detta resulterar många gånger i att växterna dör och behöver slängas bort.

Därför kom idén om ett automatiskt bevattningssystem till krukväxter.

Genom att ta fram en prototyp som vattnar krukväxter endast när de behöver vatten, kan flera problem elimineras. När växterna får vatten i den mängd de önskar så ökar livslängden på dem och på så sätt kan de fortsätta frodas. Till följd kommer mängden växter som slängs minskas.

1.2 Syfte

Syftet med detta projekt är skapa en prototyp för ett system som ska vattna krukväxter baserat på fuktigheten i deras jord. Tydligare mål och begränsningar finns i sina respektive rubriker **1.3 Avgränsningar** och **1.4 Precisering av mål**.

1.3 Avgränsningar

För att kunna testa många olika typer av scenarion kommer tre olika krukväxter med olika behov att testas: Fredskalla, Gullranka och Orkidé, se mer information i avsnitt

2.4. Projektet har en begränsad budget, därför kan kvalité och utseende variera mot ifall bättre komponenter införskaffats. Projektet utförs med en begränsad tid på 400 h. Förslag på förbättringar och vidareutveckling av prototypen diskuteras i en egen del i sista avsnittet **5 Slutsats och Diskussion**

1.4 Precisering av mål

Projektet styrs av satta mål där huvudmålet och slutmålet är en prototyp till en bevattningsanordning för tre krukväxter. Den i sig ska endast vattna när plantorna behöver vatten och systemet ska veta detta genom att mäta fuktigheten i jorden och temperaturen i luften med hjälp av olika typer av givare.

Det finns olika områden som arbetas med i projektet: programmering, koppling av elkomponenter, bygge av prototyp och att få alla delar att kommunicera med varandra. Varje område har sina delmål. Dessa presenteras nedan:

Programmering:

- Funktion som ger mycket eller lite vatten.
- Funktion som kan låta växterna torka mellan bevattning.
- Funktion för att ändra börvärde.
- Två olika regulatorer.
- Funktion som minskar periodtiden om det är varmt.
- En säkerhetsfunktion.

Koppling av elkomponenter:

- Koppla på ett lättåtkomligt sätt.
- Färgkoda kablarna för att underlätta felsökning.
- Koppla säkert.

Bygge av bevattningssystemet:

- Prototypen ska vara så billig som möjligt.
- Prototypen ska se estetiskt bra ut men ändå vara lätt att kunna felsöka på.

Kombinering av alla delområden:

- Kunna bestämma vilken bevattningsslang som inställningarna ska ändras på genom att trycka på en knapp.
- Kunna låta användaren välja om plantan ska torka ut jorden mellan bevattningar.
- Kunna bestämma om varje blomma ska få en större eller mindre mängd vatten när den vattnas.
- Kunna bestämma fukthalten varje blomma ska ha genom att ändra med en vridpotentiometer.
- Kunna visa på displayen inställda fukthalten på varje blomma.
- Kunna visa på displayen vad nuvarande fukthalt är för varje växt.
- Kunna visa omgivningstemperaturen på LCD genom att vrida på vridpotentiometern när användaren är på "startsidan".
- När vattennivån är för låg ska en givare se till att pumpen inte startar och visa detta med text på LCDn.
- Den ska stoppa bevattningen ifall pumpen har varit på för länge.
- Om temperaturen är över en viss grad ska den vattna oftare.

1.5 Arbetsmetod

Nedan står metoden som användes för att utföra och färdigställa projektet.

- Litteraturstudier genomfördes.
- Informationssökning om Arduino Uno samt påläsning och test för att lära sig utvecklingsmiljön Arduino IDE.
- Information om de valda växter inhämtades från olika hemsidor om blommor, hur de ska vattnas, hur mycket vatten och i vilken miljö de ska vara.
- Diskussion angående hur systemet ska se ut, vilka komponenter som kan tänkas användas och hur många av varje komponent.
- Därefter diskussion om de inköp som behöver göras.
- Konstruktion av alla komponenter med programmering parallellt.
- Test av programkod för att få fram olika parametrar. Det togs också fram genom informationssökning.
- Tester av tidsparametrar för pump och magnetventil. Programmet kördes och mätning av vad som kändes vara lagom mycket vatten till en blomma bestämdes.

2 Teori

Nedan följer den teoretiska delen som förklarar ord och begrepp som används i rapporten och vad dessa är.

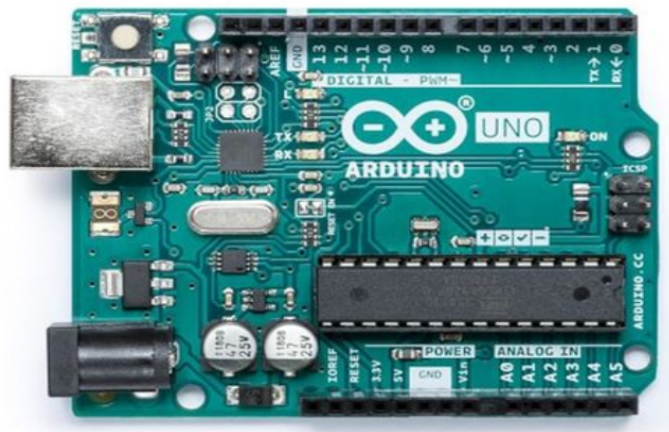
2.1 Arduino[®]

Arduino[®] är en typ av utvecklingskort med öppen hårdvara. Märket har många olika varianter där den mest använda är Arduino[®] Uno [1]. En Arduino[®] Uno består av en mikrokontroller och flera olika elektronikkomponenter. Med den kan många olika typer av funktioner skapas. Allt från att få en lampa att blinka till att styra en liten robot.

När utvecklingskortet ska programmeras används utvecklingsmiljön Arduino[®] IDE. Den laddas ned på en dator, och för att få utvecklingskortet att kommunicera med datorn används en USB-A till USB-B kabel. Språket i Arduino[®] IDE liknar på många sätt programmeringsspråket C, men är en mer förenklad variant.

Arduinon[®] drivs bäst med en inspänning på 7-12 V, men kan även fungera med 6-20 V. Antingen så kan den matas med spänning från datorn genom USB-kabeln, eller så finns det möjlighet att koppla in direkt till ett vägguttag. En Arduino[®] i sig drar väldigt lite ström, men beroende på vad som händer i (och runt omkring) så kan den dra mer eller mindre [2]. Den maximala spänningen som kommer från Arduinon[®] är 5 V men det finns också en 3,3 V utgång. Det går att lagra data då den har ett flashminne på 32kB [1].

På Arduino[®] Uno finns mikrokontrollern ATmega328. Den har 28 ben och är en 8-bitars MCU. Det finns 6 analoga ingångar och 14 digitala in- eller utgångar. Där- efter finns det två jordade ben och ett där 5 V ges ut och ett annat där 3,3 V är utspänningen. Det finns ett ben som heter SCL och ett som heter SDA, de används till en så kallad kommunikationsbuss. Enkelt beskrivet så används detta till att koppla ihop olika komponenter så de kan kommunicera med varandra, beskrivs lite mer i **2.2.3 LCD**. Nedan, figur 2.1, syns en bild på en Arduino[®] Uno.



Figur 2.1: Arduino[®] Uno [3]

Skillnaden på att använda en Arduino[®] med en MCU inbyggd och enbart använda en lös MCU är att den i Arduinon[®] är redo att användas direkt men en lös MCU kräver förarbete innan den kan programmeras. De har olika utvecklingsmiljöer, Arduino[®] använder sig av Arduino[®] IDE och MCU använder sig av annan utvecklingsmiljö, exempelvis MPLAB XC8. Arduino[®] har inbyggd Rx, receiver (mottagare) och Tx, transmitter (sändare). Rx och Tx är nödvändigt för att datorn ska kunna skicka data till Arduinon[®] och viceversa.

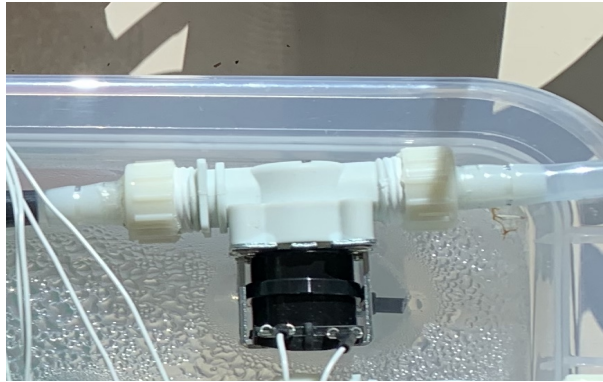
2.2 Beskrivning av elektronikkomponenter

De elektronikkomponenter som användes för att bygga ihop systemet står beskrivna nedan.

2.2.1 Magnetventiler

Magnetventiler är en typ av ventiler som används till att bland annat öppna och stänga olika flöden. Det kan vara vätskor eller gaser. Det finns två typer av magnetventiler, de som normalt är stängda och de som normalt är öppna [4]. De fungerar så att när det kommer en viss spänning skapas ett magnetfält som öppnar den ventilen som normalt är stängd och stänger den som normalt är öppen och viceversa när

spänningen försvinner. Nedan i figur 2.2 syns en bild på en magnetventil.



Figur 2.2: 12 V magnetventil.

2.2.2 Universal spolarpump

Pumpen är en vindrutespolarpump och används vanligtvis i bilar [5]. Nedan i figur 2.3 syns en bild på pumpen. Den går oftast på 12 V där pumpen kopplas från en strömkälla till en plus- och minuspole på pumpen. Strömkällan måste kunna ge tillräckligt mycket ström som exempelvis ett bilbatteri. Denna pump används för att pumpa vätska.



Figur 2.3: Vindrutespolarpump [6].

2.2.3 LCD

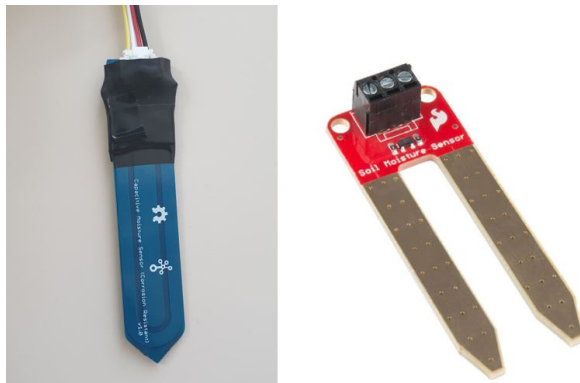
En LCD är en enklare typ av skärm som består av pixlar som antingen kan vara på eller av. Dessa styrs genom att skicka in olika digitala signaler som ger önskad funktion, exempelvis skriva ut olika saker på skärmen. LCD kan använda sig av kommunikationsbussen I2C. Bussen I2C används av många komponenter då den är enkel att implementera, den går även att inkludera till Arduinon[®]. Den använder sig av två linjer eller leder, SDA och SCL. SDA tar med sig data och SCL är en klocksignal som kan synkronisera dataöverföringen mellan olika enheter på kommunikationsbussen [7].

2.2.4 Strömställare / Strömbrytare

Det finns flera olika varianter av strömställare och strömbrytare. De finns som tryckknapp eller switch och deras funktion är att sluta eller bryta en krets.

2.2.5 Fuktighetsgivare

Denna elektronikkomponent är till för att mäta fuktigheten i jorden eller luften. De primära varianter som finns är resistiva och kapacitiva. De resistiva brukar bestå av två ben där resistansen mäts mellan de två benen och på så sätt avgörs hur fuktigt det är. Kapacitiva fuktighetsgivare fungerar genom att mäta de kapacitiva förändringarna i kondensatorn som givaren består av. Dessa förändringar sker när elektrikum (membranet mellan kondensatorplattorna) påverkas av vattnets joner i jorden[8][9]. Nedan i figur 2.4 syns en kapacitiv och en resistiv givare.



Figur 2.4: Vänster: en kapacitiv givare. Höger: resistiv givare [10].

2.2.6 Temperaturgivare

En temperaturgivare kan antingen vara analog eller digital. Det finns många olika sorter. Något som exempelvis kan skilja två givare är vilket temperaturintervall de kan ha. Vissa givare kan exempelvis mäta från 0 till 100 °C och andra kan istället mäta från -40 till 125 °C. De kan också tåla och/eller drivas på olika mycket spänning. En annan skillnad mellan en digital och analog temperaturgivare är att de måste kopplas till olika ben på Arduinon[®] beroende på vilken som är en analog port och vilken som är digital.

Det finns flera sätt för temperaturgivare att mäta temperaturen, exempelvis en resistanstermometer och ett termoelement. En resistanstermometer mäter temperaturen genom att mäta resistansändring i givaren. Ett termoelement mäter den temperaturen genom att omvandla temperaturskillnaden mellan referenstemperaturen (som oftast rummets temperatur) och den uppmätta till en spänning [11].

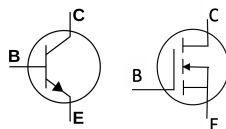
2.2.7 Vridpotentiometer

En potentiometer är en elektronikkomponent som kan reglera spänningen i den delen av kretsen den finns. Detta genom att vrida den åt olika håll. Om den är vriden maximalt åt ena hållet fås ingen resistans och maximalt åt andra hållet ger högsta möjliga resistans.

Det finns två typer av potentiometrar. Antingen är de logaritmiska eller linjära [12]. En linjär fungerar som det låter, den ökar/minskar resistansen linjärt med hur mycket som vrids. En logaritmisk ändras mycket mer drastiskt i början och avtar ju mer det har vridits. Det finns olika sorters potentiometrar. Antingen kan de vridas ett eller flera varv. De finns dessutom med olika stora motstånd, allt från några 100 ohm till 100k ohm.

2.2.8 Transistor

Transistorer används som strömbrytare, signalförstärkare och spänningsreglerare. De två huvudgrupperna av transistorer är Bipolärtransistor (BJT) och Fälteffekttransistor (FET). Skillnaden mellan dessa två är att FET är spänningsstyrd medan BJT är strömstyrd [13]. En transistors uppbyggnad kan se ut på olika sätt. Det är skillnad på BJT och FET men det är också skillnad på hur de ser ut inom dessa huvudgrupper. I figur 2.5 syns elektronksymbolerna för en typisk BJT och en MOSFET, MOSFET är den vanligaste typen av FET [13]. Emmitter representeras av E, kollektor av C och bas (styre) av B. Bas används i anknytning till BJT och styre används i anknytning till FET [13].



Figur 2.5: Vänster: BJT, höger: MOSFET

I figuren ovan syns bland annat en MOSFET, där kanalen mellan emitter och kollektor är isolerad från styret. Kanalen och styre får kontakt genom att det blir en spänning mellan styre och substrat, där substratet ligger mellan kanal och styre [14]. Substrat är en molekyl som kan skapa en kemisk reaktion, som då sker när spänning uppkommer mellan styre och substrat. En annan vanlig typ av FET är JFET, den är likadant uppbyggd som en MOSFET men har inget mellanrum mellan styre och kanalen mellan emitter och kollektor.

2.3 Beskrivning av signalbehandling: AD-omvandling

Olika elektronikkomponenter använder sig av olika typer av signaler för att kunna fungera på önskat sätt. Därför behöver de signalanpassas så att två komponenter som inte använder samma typ av signal kan fungera ihop. Den signalanpassning som Arduino[®] utför är AD-omvandling. Det används när en analog signal ska omvandlas till digital. Analoga signaler kan anta alla värden medan digitala signaler kan bara anta vissa där begränsningen är vilket antal bitar systemet har. I Arduino[®] IDE finns en funktion som heter "MAP" som sköter AD-omvandlingen.

2.4 De valda växterna

Fredskalla är en växt som vill ha mycket vatten, den tål även stå i vatten och bli genomblöt. Gullranka vill ha mycket vatten men vill även torka ut mellan bevattningstillfällena. Orkidéer vill torka ut helt och sedan få vatten men inte så att den blir övervattnad. I figur 2.6 syns en bild på dessa tre blommor.



Figur 2.6: Från vänster: Fredskalla, Gullranka, Orkidé

3 Genomförande

För att få en fungerande prototyp var det nödvändigt att programmera och koppla. Här beskrivs vilka komponenter som valdes och varför.

3.1 Förarbete

Vid start av projektet ägnades det tid åt informationssökning för alla delar av projektet. Under denna period bestämdes vilka komponenter som skulle vara mest optimala för prototypen. Ett flödesschema för programmeringen skapades för att lättare kunna realisera tankar och önskemål.

3.2 Tidsplan

Efter förarbetet kunde en tidsplan skapas. Detta blev ett Gantt-schema, se **Bilaga 1**, mer specifika delmål finns i avsnittet **1.4 Precisering av mål**.

3.3 Fuktighetsgivare

Val av fuktighetsgivare stod mellan två typer: resistiva och kapacitiva. Anledningen att den resistiva föll bort i urvalet under förarbetet, var på grund av dess dåliga motstånd mot korrosion och därför skulle komma att passa dåligt till detta projekt. Den kapacitiva skulle därför passa mycket bättre då själva sensorn inte behövde sitta i jorden. Sensorn sitter skyddad innanför det yttre höljet på givaren, och på en resistiv sitter sensorn utanpå och därmed oskyddad. Kapacitiva givare fanns i många alternativ men valet föll på en givare som fungerar bra ihop med Arduino[®], då den går på 3,3 V och är analog [15].

3.4 Temperaturgivare

Arduino® Uno har få analoga ben att koppla komponenter till. På grund av detta valdes därför den digitala temperaturgivaren DS18B20.

En anledning att mäta temperaturen nära växterna är att vissa plantor trivs bäst vid en viss temperatur och en givare kan då ge indikation ifall den inte står i optimala förhållanden. Mer detaljerat finns i avsnitt **3.14.3 Hög temperatur**. Om det är en växt som vill ha det varmare kan temperaturindikationen på LCDn hjälpa att bestämma om växten behöver flyttas.

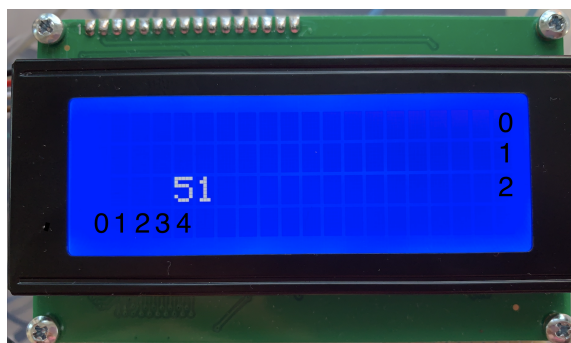
3.5 LCD

För att lätt kunna ha översikt över fukthalten i jorden och temperaturen så koplades en LCD in till systemet. Den valda LCDn har plats för 20 tecken och fyra rader, eftersom det var det som skulle passa bäst enligt planering i förarbetet. Användningen av LCDn var inte komplicerad då det enbart krävdes några rader kod för konfiguration av skärmen. Sedan för att skriva ut något på den var det enbart att göra enligt figur 3.1.

```
392 |         lcd.setCursor(4, 2);  
393 |         lcd.print(arvarde2);
```

Figur 3.1: Kod till LCD.

I figuren betyder "lcd.setCursor" vart i LCDn det ska skrivas ut, här betyder det fyra steg från vänster med start på 0 och två steg uppifrån. "lcd.print" betyder vad som ska skrivas ut, första bokstaven/siffran börjar på plats fyra i detta fall, se figur 3.2. "arvarde2" blev i detta fall 51, vilket betyder 51 % fukt i jorden.



Figur 3.2: LCD som visar vad figur 3.1 skriver ut.

LCDn kopplas till jord och 5 V. Sedan finns det två portar till, ena är SCL och den andra är SDA, beskrivet i avsnittet **2.2.2 LCD**. De kopplas till varsin analog port på Arduinon[®], A4 och A5.

3.6 Pump

För att transportera vattnet från vattenbehållaren till växterna behövs en pump. En vanlig pump drivs ofta genom ett vägguttag. Då det är höga strömmar i ett vägguttag och Arduinon[®] inte klarar av det, lämpar det sig bättre med en pump som kan drivas med ett batteri istället. Detta löstes med en universal spolarpump. De kan finnas i t.ex bilar. Anledningen till att det var just en spolarpump var för att den kunde kopplas in i en vanlig krets eftersom den går på 12 V.

Den valda pump har en innerdiameter på 4,5 mm. Den drivs på 12 V och drar 3,33 A. Den syns i figur 2.3 i avsnitt **2.2.2 Universal spolarpump**.

3.7 Nivågivare

Nivågivare används till bevaka vattennivåer. I figur 3.3 syns flottören (den vita) som sitter på givaren ger en signal när den kommit i botten. När den flyter upp igen med vattennivån ger den en ny signal. Vad som sker vid dessa två olika signaler måste programmeras.



Figur 3.3: Nivågivare [16].

Vattenpumpar får inte torrpumpas en längre period eftersom att de då överhettas och går sönder. Det fanns risk för detta när vattennivån i behållaren blev låg. En nivågivare var bra då den skulle varna via text på LCDn och stänga av pumpen ifall det var lågt vatten i behållaren.

För detta projekt var det bara intressant att veta huruvida det var tillräckligt med vatten i behållaren eller inte. En digital nivågivare valdes på grund av att den kommer bara mäta två värden, av eller på, och det finns fler digitala portar på Arduino[®] än analoga. Denna behövde endast kopplas till jord och till en port på Arduinon[®] som har en spänning på 5 V.

3.8 Magnetventiler

Magnetventilerna användes till att bestämma vilken växt som skulle få vatten, och fungerade som en extra säkerhetsåtgärd mot vattenläckage och eventuella problem. De som valdes är ventiler som normalt är stängda. Detta innebär att skulle strömmen brytas stängs ventilerna och då kan det inte rinna ut vatten.

Valda ventiler klarar av en spänning på max 12 V, och öppnades efter tester runt 6 V. In- och utflöde på ventilerna är 8 mm i diameter.

3.9 Kombination av pump och magnetventiler

För att spolarpumpen och magnetventilerna skulle kunna transportera vattnet ihop behövdes ett flertal komponenter. Till och från pumpen kopplades slang med 5 mm i innerdiameter. Kort efter sattes en X-koppling, vilket är en komponent som kan koppla ihop flera slangar och har fyra in-/utgångar [15]. Magnetventilerna som skulle kopplas ihop med denna X-koppling hade en större inne- och utdiameter, den var på 8 mm. Därför behövdes tre reduceringskopplingar. En reduceringskoppling används till att koppla ihop slangar som har olika storlekar. Det behövdes reducering från 8 mm till 5 mm, men då detta inte fanns fick det bli 8 mm till 6,3 mm [18]. Sist kunde 8 mm slang kopplas på som sedan kopplas till magnetventilerna och sedan vidare till växten.

3.10 Drivning av prototypen

Under teststadie och framtagning drevs Arduinon[®] av en dator och de andra komponenterna av ett spänningsaggregat. Detta skulle inte vara hållbart i längden och därför införskaffas verktyg så att bevattningssystemet skulle fungera på egen hand. Det realiserades genom att magnetventilerna och pumpen drevs av ett laddningsbart batteri med en spänning på 12 V och en livslängd på 7,2 Ah. Arduinon[®] drevs i slutändan bara av en dator men hade kunnat drivas av en batterieliminatör.

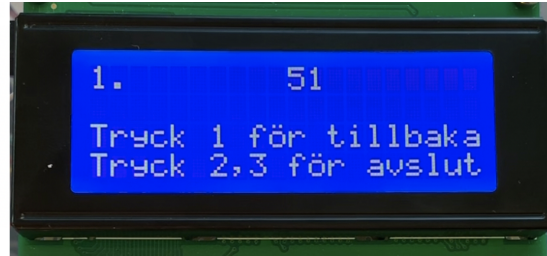
3.11 Elektronikkomponenter

Det användes ett flertal elektronikkomponenter, de valda varianterna beskrivs nedan och även hur de användes.

3.11.1 Potentiometer

För att kunna ställa in börvärden och även kunna se temperaturen användes en vridpotentiometer. Detta för att den skulle vara mer användarvänlig. Inställning av börvärden går enbart att ändra när programmet befinner sig i rätt funktion, se figur 3.4.

```
555         while(varde1==1){
556             lcd.setCursor(0, 0);
557             lcd.print("1.");
558             lcd.setCursor(0, 2);
559             lcd.print("Tryck 1 för tillbaka ");
560             lcd.setCursor(0, 3);
561             lcd.print("Tryck 2,3 för avslut");
562             potentiometer1();
563         }
```



Figur 3.4: Ändring av börvärdet. Vänster: programkod för att ändra börvärde. Höger: vad LCD visar vid ändring av börvärde

Där går det att sätta börvärdet till en procent mellan 0 och 100. Potentiometern omvandlar värden i en annan funktion där den omvandlar de värden den får in genom "MAP", se **Bilaga 2**. Om LCDn befinner sig på startsidan så kommer det inte gå att ändra börvärde, däremot när potentiometer är vriden till max, kommer allt som står i LCDn försvinna och temperaturen visas.

3.11.2 Strömbrytare / knappar

För att lättare kunna navigera prototypen på LCDn användes strömbrytare i form av knappar. Dessa användes för att ändra inställningar vid till exempel ändring av den fukthalt varje växt skulle ha. Antal knappar bestämdes efter hur många växter det skulle vara, alltså tre. Dessa knappar har mer än en funktion. De ska bland annat kunna användas för att komma fram till ändring av inställningar. Det står alltid beskrivet på LCDn vilka knappar som ska tryckas på.

3.11.3 MOSFET

Komponenterna som användes i projektet behövde olika mycket spänning för att drivas och därför sattes en MOSFET in. Transistorns funktioner i prototypen var att se till att magnetventilerna och pumpen kunde kommunicera med Arduinon[®]. Både pumpen och ventilerna krävde 12 V men Arduinon[®] klarar endast att ge ut max 5 V samtidigt som den endast tål lite ström. Den transistor som användes var då TIP120 med en N-kanal, se figur 3.5, den tål 60 V och 5A vilket räckte till detta system [19].



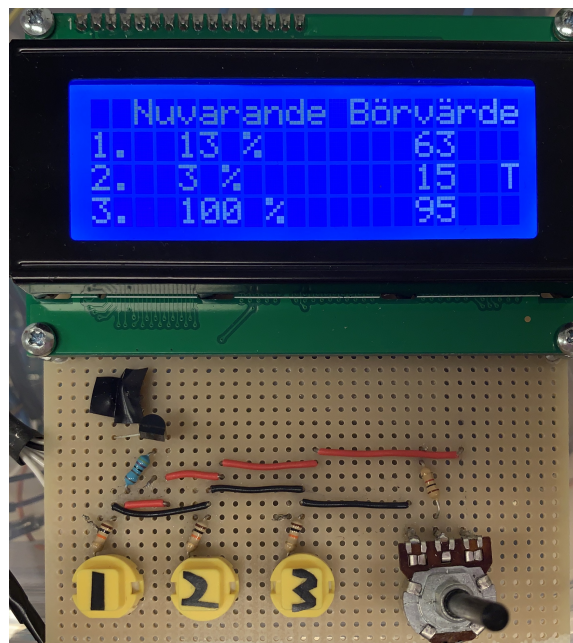
Figur 3.5: MOSFET, TIP120

Nummer 1 i figuren ovan är styre, kollektorn 2 (vilket är på två ställen) och emittern 3.

3.12 Koppling av el-komponenter

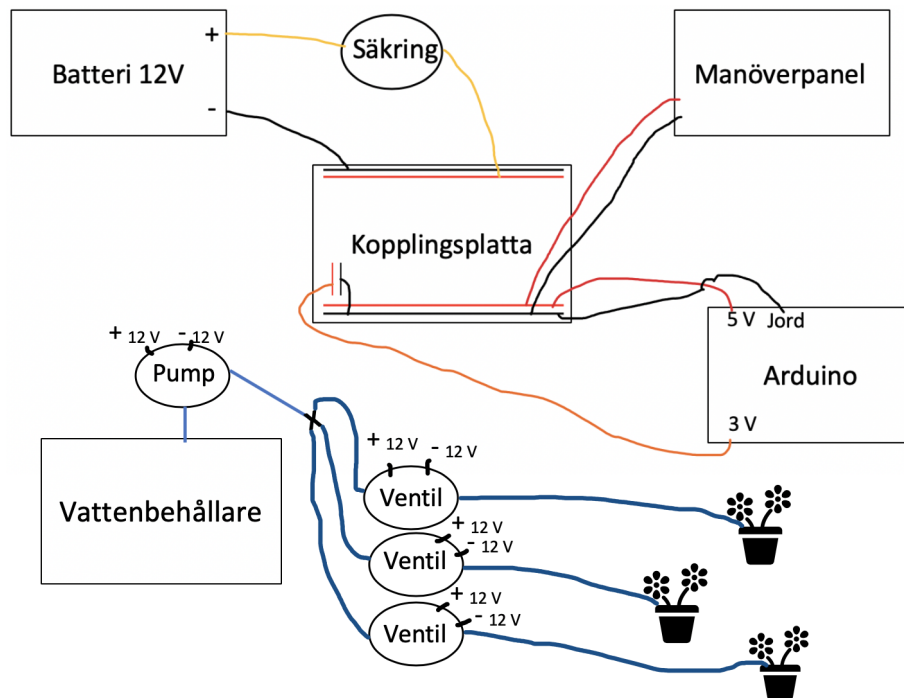
Den inköpta kopplingsplattan var primärt där alla komponenter kopplades och testades. Det var en liten kopplingsplatta så under delar av projektet användes en större variant för att lättare kunna ha översikt över alla komponenter, men också lättare kunna felsöka och ändra det som eventuellt var skadat. När detta var färdigt löddes en manöverpanel med knappar, potentiometer och en temperaturgivare. På manöverpanelen löddes även de resistorerna som behövdes och flertrådskablar som kopplades in i Arduinon[®]. Fördelen att välja flertråds kabel var att den gick att böja utan att

påverkas, då fuktighetsgivarna sitter långt från Arduinon[®] kommer kablarna att röra på sig, även manöverpanelen sitter lite längre ifrån Arduinon[®]. En entråds kabel är stel och kan då riskera att gå av om den böjs för många gånger. Till lödningen av manöverpanelen behövdes kablar mellan komponenter och då användes entråds kabel eftersom den ska ligga still. Se figur 3.6 för framsidan av manöverpanelen, där syns även entrådskablar i svart och rött.



Figur 3.6: Manöverpanel.

För att fuktighetsgivarna, pumpen och magnetventiler skulle få ström användes den inköpta kopplingsplattan. På den kopplades 12 V direkt från batteri. Från pluspolen av batteriet sattes en säkring in före den kopplades till plattan, den låg på 5A. I figur 3.7 syns en skiss på kopplingsschemat över delar av prototypen.



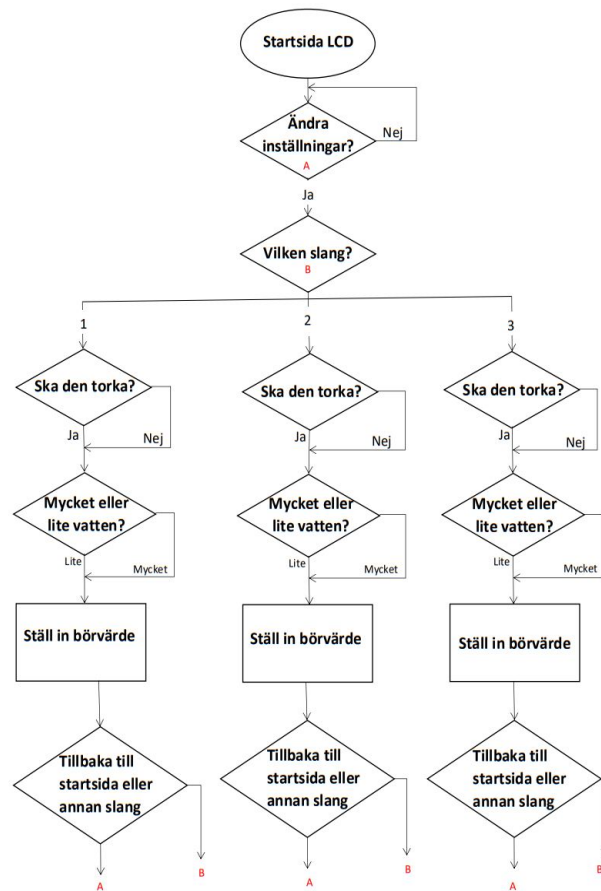
Figur 3.7: Skiss över prototypens delar.

Pump och ventiler är kopplade till fler komponenter än bara spänningskällan 12 V, alltså mer än vad figuren visar. De kopplas från kopplingsplattan vidare till Arduinon[®] också. Exakt vart alla komponenter kopplas kan ses i **Bilaga 3**.

3.13 Programbeskrivning

Programmeringen delades upp genom att funktioner skapades till varje elektronikkomponent som behövde styras. När dessa fungerade så kombinerades allt ihop och flera mer komplicerade funktioner kunde skapas.

Programmeringen gjordes efter flödesschemat, se figur 3.8 nedan, som togs fram i förarbetet. I den är fokus på vad användaren kan tänkas vilja ändra och se på LCDn.



Figur 3.8: Flödesschema av programmet.

I figuren ovan syns, efter val av slang, frågan "Ska den torka?". Det går då att välja Ja eller Nej. Vad än valet är kommer programmet komma ihåg inställningen och gå vidare till samma fråga vilket är "Mycket eller lite vatten?". Där är det samma princip som för ska den torka, den kommer ihåg vad valet är och går vidare till nästa steg vilket är här "Ställ in börvärde".

3.14 Regulatorer

I förarbetet undersöktes flera olika avancerade typer av regulatorer för att se vilken som skulle passa bäst till detta system. Att välja en regulator var viktigt så den skulle vara så effektiv som möjligt. De regulatorer som kom på tal var PI reglering, Fuzzy Control och tvålägesregulator. Efter samtal med handledaren Bertil Thomas eliminerades Fuzzy Control då det skulle bli för komplicerat att skapa och PI reglering eliminerades eftersom det skulle bli onödigt avancerat.

Regulatorn som valdes blev av typen tvåläge och programmerades i Arduino® IDE. En tvålägesregulator har två tillstånd: på eller av. Vilket tillstånd den ska ha beror på hur programmeringen är gjord, där jämförelser av värden sker och på så sätt bestämma om den ska vara på eller av. Detta sker alltså varje gång programmet går eller vid vissa bestämda periodtider.

Två olika regulatorer används i projektet: en vanlig och en för om det ska torka ut mellan bevattningarna, vilka kommer beskrivas nedan. Dessa i sin tur finns i två olika varianter: om det skulle vara för varmt och om de ska ha mycket vatten eller inte, även de finns beskrivna nedan. Den vanliga regulatorn och torkningsregulatorn kan fungera parallellt, exempelvis kan en växt ha inställningen att den ska torka och de andra två har inställningen vanliga regulatorn. Regulatorerna och de olika varianterna kommer beskrivas nedan och tidskonstanter som används kommer förklaras i **3.15 Test av prototyp**.

3.14.1 Vanlig regulator

Den vanliga regulatorn är inprogrammerad så den är på när Arduinon® startas första gången, men det går även att göra den inställningen senare när programmet varit på ett tag. Då går det att ta på den när inställningarna ändras och LCDn frågar ”Ska den torkas ut?” och svaret är NEJ. För att undvika att växterna ska övervattnas vattnar den bara i 5, 3 eller 2 sekunder vid lite vatten (exakta tider beskrivs i tabell 1) och därefter väntar i 1 timme, vilket är periodtiden, innan programmet kollar igen om det behövs mer vatten. Skulle temperaturen vara över 35°C kommer periodtiden att halveras. Om fukten är under sitt börvärde kommer den vattna i en viss bestämd tid och upprepa denna procedur tills börvärdet är nått. Om fukthalten i jorden är över sitt börvärde kommer inte magnetventilen att öppna sig när programmet kontrollerar.

Anledningen till att den inte vattnar tills fuktprocenten är nådd på en gång är för att då finns det en risk att det blir övervattnat, eftersom jorden inte hinner ta upp allt vatten. Fuktgivaren mäter enbart fukten runt själva givaren och inte i hela krukan med jord, det leder till att om vattnet enbart hamnar i jorden där givaren inte sitter så kommer den inte att märka det och växten kommer bli för blöt.

3.14.2 Torkningsregulator

När inställningar ändras kommer programmet fråga om växten ska bli torr mellan bevattningarna, om svaret är JA kommer programmet hoppa in i en egen funktion, torkningsregulatore. Det funktionen gör är att när en växt har fått inställningen att den ska torka mellan bevattningarna så kommer den vänta med att starta pump och magnetventil tills fukthalten i jorden är under 8 %, vilket räknas som tillräckligt torr. När fukthalten då gått under 8 % kommer den fungera som den vanliga regulatore med att vara öppen i 5, 3 eller 2 sekunder och sedan efter 1 timme kolla om mer vatten behövs och då upprepa processen med att starta om cykeln. Är den valda fuktprocenten i jorden lika med eller över den verkliga fukthalten i jorden vid nästa kontroll kommer pump och magnetventil stängas ända tills fuktprocenten gått under 8 % igen. För att se på LCDn om en växt ska torka betecknas det med ett "T" längs till höger bredvid börvärdet.

3.14.3 Hög temperatur

Hög temperatur är som nämnt en variant av regulatorerna som har skapats. Det finns en temperaturgivare på manöverpanelen som mäter temperaturen hela tiden. När det blir för varmt för en växt så behöver den mer vatten, eftersom vatten avdunstar snabbare vid höga temperaturer. Denna variant fungerar så att när det blir över 35°C kommer den följa samma sätt som för de nämnda regulatorerna men tiden mellan kontrollerna, periodtiden, är halverad. Därför kommer den kontrollera varje halvtimme. Valet av varm temperatur, 35°C, kommer från egna antaganden.

3.14.4 Mycket eller lite vatten

Den andra varianten är om en växt ska ha mycket eller lite vatten. Exempelvis om det skulle vara en stor kruka med mer jord vill en växt ha mycket vatten och om det skulle vara en liten kruka vill kanske växten ha lite vatten. Bevattningstiderna, tiden som pumpen och ventilen är på, är det enda som ändras om det ska vara mycket vatten. Tiderna beror på hur många växter som behöver vatten. De finns beskrivna i tabell 1. Alla tre växter behöver inte ha samma inställning av mycket eller lite vatten, utan de kan fungera parallellt. Exempelvis kan två växter vilja ha mycket vatten och en vilja ha lite vatten. För att se från LCDn vilka som behöver mycket eller lite vatten betecknas mycket vatten med ett ”+” och lite vatten med ett mellanslag, alltså ” ”.

Tabell 1: Bevattningstider

Mycket eller lite vatten	Tre blommor behöver vatten	Två blommor behöver vatten	En blomma behöver vatten
1 st behöver mycket vatten, 2 st behöver lite vatten	1 ventil stängs efter 7 s, 2 ventiler stängs efter 5 s	1 ventil stängs efter 5 s, 2 ventiler stängs efter 3 s	1 ventil stängs efter 4 s, 2 ventiler stängs efter 2 s
2 st behöver mycket vatten, 1 st behöver lite vatten	2 ventiler stängs efter 8 s, 1 ventil stängs efter 5 s	2 ventiler stängs efter 6 s 1 ventil stängs efter 3 s	
3 st behöver mycket vatten	3 ventiler stängs efter 9 s		

För flödesschema av den vanliga regulatorn se **Bilaga 4**, för flödesschema av torkningsregulatorn se **Bilaga 5**.

3.15 Test av prototyp

Regulatorn kan byggas på olika sätt. Den kan exempelvis kolla fuktighet var 5:e minut eller varje timme. För att få rätt tid mellan varje gång programmet kollade om en växt behövde vatten behövde tester göras. I test 1 till 6 nedan är växt 1 en Gullranka, växt 2 en Orkidé och växt 3 är en Fredskalla. Test 7 till 9 spelar inte växterna någon roll då endast funktionerna testades. Mellan varje test ändrades lite olika parametrar, vilket kan ses nedan.

Test 1

En av tre fuktgivare var trasig så därför utfördes detta test endast på två växter, växt 2 och växt 3. Växt 2 vill torka ut efter att den nått sitt börvärde, ca 18 % fukthalt i jorden. Växt 3 vill ha mycket vatten, ca 70 % fuktighalt i jorden hela tiden. Alltså en växt där torkningsregulatorn var på och en där vanliga regulatorn var på. Parametrar som var ändrade här var att den skulle kontrollera var 15:e minut, periodtiden var på 15 minuter, och pumpen och magnetventil skulle vara på i 5 sekunder. Efter de första 15 minuter var växt 2 på 1-4 % fukt och växt 3 på 37-39 %. Efter 15 minuter

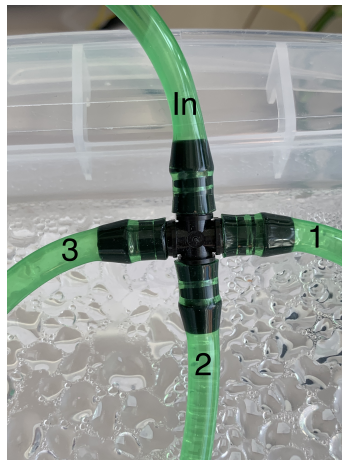
till gick fuktprocenten upp på växt 2 till 4-6 % och 39-41 % på växt 3. Mellan denna omgång och de nästa 15 minuterna gick procenten i växt 2 upp till runt 85 %, när växten då lyftes upp ur krukans sågs att den var fylld till cirka hälften vatten. Den hade alltså blivit övervattnad, men växt 3 mår fortfarande bra och fukthalten i jorden steg i stadigt tempo upp till sitt börvärde.

Test 2

Förutsättningarna var likadana, med bara två fuktgivare till samma växter som **Test 1**. Under detta test ändrades enbart att pumpen skulle vara öppen i 5 sekunder för den vanliga regulatorn och 3 sekunder för torkningsregulatorn. Då testet utfördes direkt efter det första testet var växt 2 väldigt fuktig, ca 50 %, vilket ledde till att magnetventilen öppnades inte då den inte behövde mer vatten. Men det som gick att se ur detta test var att fukthalten i jorden hos växt 3 höjdes betydligt mer och det berodde på att den fick allt vatten när de andra ventilerna var stängda. Slutsatsen som drogs var att olika tidsparametrar behövdes beroende på hur många växter som behövde vatten.

Test 3

I detta test fanns alla tre fuktgivare och alla växter behövde vatten. Först testades att pumpen och ventilerna var på i 10 sekunder. Det visade sig vara alldeles för länge. Växt 2 fick för mycket vatten, det beror antagligen på att det är den som alltid kommer få lite mer vatten. I en X-koppling strömmar mest vatten ut till växt 2 eftersom den är i rakled mot inflödet, se figur 3.9.



Figur 3.9: X-koppling

10 sekunder var för mycket så då testades 7 sekunder, det var också för mycket. Sist testades 5 sekunder, vilket kändes som en lagom mängd vatten. Det går inte att veta exakt vad som är lagom mycket vatten. Lagom mycket vatten bestämdes till ca 1 dl.

Test 4

I detta test behövde två växter vatten. Eftersom **Test 3** ledde till att 5 sekunder används om alla tre är på, kunde slutsatsen dras att det behövdes färre sekunder vid bevattning av 2 växter. 3 sekunder testades och det kändes som en rimlig mängd vatten som kom fram genom de två ventilerna som var på.

Test 5

I detta test behövde enbart en växt vatten. **Test 4** fick fram att om 2 växter behövde vatten skulle pumpen vara på i 3 sekunder, det ledde till att detta test testades med 2 sekunder. Den mängd vatten som kom var lagom mycket vatten.

De tidskonstanter som bestämdes var då 5, 3 och 2 sekunder för både den vanliga regulatorn och för torkningsregulatorn. Alla tider står beskrivet ovan i tabell 1.

Test 6

Test 6 kontrollerade hela systemet med de valda tidskonstanterna beskrivna ovan. Efter testet som varade i 5,5 timmar var det inte så stor skillnad i fukthalten vilket det inte ska vara i växt 3. Växt 3 höll sig runt 78% under hela testet och växt 2 höll sig runt 26%. Programmet fungerade som det skulle med hur länge pumpen och magnetventilerna var öppna.

Test 7

De 3 kommande testerna undersökte funktionen som gav mycket vatten. Test 7 gick ut på att testa funktionen med den vanliga regulatorn. I tabell 2 kan varianterna som testas läsas av. Där står "V" för Vanliga regulatorn. "T" står för Torkningsregulatorn. "+" står för mycket vatten.

Tabell 2: Varianter vid test: Mycket vatten - Vanlig regulator

Blomma 1	Blomma 2	Blomma 3
V	V	V
V	V	V+
V	V+	V
V	V+	V+
V+	V	V
V+	V	V+
V+	V+	V
V+	V+	V+

Resultat av detta test var att det kom den mängd vatten till varje växt som var önskad. Det betydde att den vanliga regulatorn och mycket och lite vatten fungerade ihop.

Test 8

Detta test var som föregående test men då användes endast torkningsregulatorn. I tabell 3 syns de försöken som utfördes.

Tabell 3: Varianter vid test: Mycket vatten - Torknings-regulator

Blomma 1	Blomma 2	Blomma 3
T	T	T
T	T	T+
T	T+	T
T	T+	T+
T+	T	T
T+	T	T+
T+	T+	T
T+	T+	T+

Resultatet av detta test var att rätt mängd vatten kom till varje växt. Torkningsregulatorn och mycket och lite vatten fungerade alltså ihop.

Test 9

Sista testet var en kombination av test 7 och 8. I detta test så undersöktes om båda regulatorerna kunde köras samtidigt med funktionen som gav mycket vatten. Alltså blandades försöken med att använda sig av den vanliga regulatorn och torkningsregulatorn och sedan växla mellan att behöva mycket eller lite vatten. I tabell 4 kan de varianter av försök som utfördes avläsas.

Tabell 4: Varianter vid test: Torkningsregulator och Vanlig regulator

Blomma 1	Blomma 2	Blomma 3
T+	V+	V+
V+	T+	V+
V+	V+	T+
T+	T+	V+
T+	V+	T+
V+	T+	T+
T	V	V
V	T	V
V	V	T
T	T	V
T	V	T
V	T	T
T+	V	V
V	T+	V
V	V	T+
T+	T+	V
T+	V	T+
V	T+	T+

Resultatet av detta test var att rätt mängd vatten kom till de tre växterna. Efter test 9 bedömdes att prototypen fungerade som önskat och därmed avslutades testfasen.

3.16 Säkerhetsfunktion

Programkoden för pumpen och ventilerna är testad många gånger när allt har fungerat bra, men det går aldrig att vara 100 % säker på att det kommer gå rätt varje gång. Därför har en säkerhetsfunktion tillämpats. I figur 3.10 går det att se koden till säkerhetsfunktionen.

```
838 void safety(){
839   if (knappstatus==LOW){
840     lagtatten();
841   }
842
843   if((digitalRead(11) == HIGH)){
844     if(millis() >= (raknarepump+11000)){
845       lcd.clear();
846       while(1){
847         stoppump();
848         lcd.setCursor(3, 1);
849         lcd.print("SÄKERHETSSTOPP");
850         lcd.setCursor(3, 3);
851         lcd.print("TRYCK PÅ RESET");
852       }
853     }
854   }
855   if((digitalRead(8) == LOW) && (digitalRead(9) == LOW) && (digitalRead(10) == LOW)){
856     stoppump();
857   }
858 }
```

Figur 3.10: Kod till säkerhetsfunktion

Den kollar först om vattennivån är låg, radnummer 839. Sedan kollar funktionen om pumpen varit på för länge, radnummer 843 till 854, vilket är 11 sekunder i detta fall. Om något av detta sker kommer pumpen stoppas och "SÄKERHETSSTOPP" kommer att skrivas ut på LCDn, då behövs det tryckas på reset på Arduinon[®] för att programmet ska fungera igen.

4 Resultat

Resultatet var fullt fungerande och uppfyllde alla mål som hade satts. Bevattningsanordningen kunde vattna tre olika krukväxter där inställning för varje individuell blomma kunde bestämmas på en LCD med hjälp av tre tryckknappar och en vridpotentiometer. Det gick att välja mellan två olika regulatorer, en vanlig regulator som håller det givna börvärdet och en torkningsregulator som går ned till 8 % innan den vattnar upp till det givna börvärdet. Den färdiga kopplingen av alla komponenter kan ses i **Bilaga 3**.

Projektets slutdatum blev förskjutet med ca 1 månad vid jämförelse med **Bilaga 1: Gantt-schema**. Detta på grund av en veckas sjukdom och sedan den stora påverkan som skedde när Covid-19 spred sig över världen.

4.1 Felkällor

För den slutgiltiga prototypen kan det uppkomma flera felkällor, fuktgivarna kan på grund av många anledningar hamna snett och upp på jorden och därmed inte mäta korrekt fuktighet. Slangarna kan även lossna från krukorna, då det inte finns någon säkerhetsfunktion för detta fall kommer pumpen och ventiler agera som om det inte hänt något och fortsätta tills vattenbehållaren är tom.

Funktionen i Arduino[®] som räknar och är basen till jämförelser av sekunder heter "millis". Den kan räkna i cirka 50 dagar sedan blir det overflow, därefter ska den enligt olika sidor på internet börja om på 0 igen men det är inte testat i detta projekt. Det leder till att ingen garanti på detta kan ges.

5 Slutsats och Diskussion

Vi tycker att vi har lyckats med projektet, att det blev en bra prototyp till ett automatiskt bevattningssystem. I den här sektionen dras slutsatser och resultatet diskuteras. Även eventuella förbättringar som går att göra om projektet skulle fortsätta.

5.1 Förbättringar

Eftersom detta endast var en prototyp som togs fram under tidsbegränsning har den potential att förbättras. Till en början hade kvalitén på komponenterna kunnat bättras, men också minskas. Om en pump med samma diameter på utflöden som magnetventilerna hade använts hade reduceringarna kunnat elimineras. Val av slang fungerade bra vid pumpen. De större slangarna till magnetventilen blev hårda och svårhanterliga, eftersom det var de som skulle ner i växten blev det svårt att placera de på ett bra sätt. En förbättring vore att använda samma storlek av slang som för pumpen, med antingen mer reduceringar eller magnetventiler med mindre in- och utflödesdiameter.

Hade det varit en längre period projektet höll på hade fler funktioner kunnat inkorporerats som: att samla data för att bevaka fuktigheten, exempelvis hur snabbt fukten stiger eller sjunker med nuvarande temperatur.

5.2 Hållbarhet ur miljöperspektiv

Som nämnts i inledningen är det vanligt att växter torkar ut och dör om de står för länge utan vatten, vilket gör att de behöver slängas och ersättas med nya. Med detta projekt kommer detta inte vara ett problem längre, då den kommer vattna när växten behöver vatten, detta kan i sin tur leda till att användningen av vatten kan minska då risken för övervattning troligtvis är mindre.

Elförbrukningen är låg då det enbart är en pump som drar 3,33 A och magnetventiler som drar 1 A. Dessa drivs av ett batteri som har 7,2 Ah och är på i max 9 sekunder en gång i timmen, men det kommer troligtvis inte hända varje timme. Arduinon[®], som är väldigt strömsnål, drivs av ett vägguttag eller en dator.

5.3 Slutsats

Det har varit många oväntade problem med både komponenter och programmering, exempelvis inköp av komponenter var väldigt svårt då det var svårt att veta vad exakt som behövdes köpas. De oväntade problemen med programmeringen var att vi skulle använda funktionen "millis" som finns beskrivet i rapporten, vilket försvårade det hela.

Som nämnt ovan så tycker vi att projektet blev lyckat och att det har varit väldigt lärorikt ur olika synpunkter. Exempelvis att lägga upp ett projekt som är i denna omfattning utan tydliga och givna riktlinjer, att koppla el-komponenter med säkerhet i åtanke, att köpa komponenter som kan kopplas ihop och även att programmera i Arduino[®] IDE, det har varit några delar som vi lärt oss längs projektets gång.

Källförteckning

- [1] *Overview*, Sverige: Elfa, u.d. [Online]. Tillgänglig: https://www.elfa.se/Web/Downloads/_t/ds/uno_eng_tds.pdf?pid=11038919, hämtad: 15-02-2020
- [2] *Arduino*, Sverige: Kjell & Company, 2017. [Online]. Tillgänglig: <https://www.kjell.com/se/kunskap/hur-funkar-det/arduino/grundlaggande-elektronikstromforsorjning-av-arduino>, hämtad: 21-03-2020
- [3] *Arduino Uno*, Arduino, u.d. [Online]. Tillgänglig: <https://store.arduino.cc/arduino-uno-rev3>, hämtad 28-04-2020
- [4] *2-Way/2 Position valves*, 2013: Internet Archive, u.d. [Online]. Tillgänglig: <https://web.archive.org/web/20131029201311/http://www.controlandpower.com/catalog/PDFs/ASCO/ASCO%20351%20General%20Service%202-Way%20Valves.pdf>, hämtad: 20-02-2020
- [5] *Vindrutespolarpump, universal*, Sverige: Biltema, u.d. [Online]. Tillgänglig: <https://www.biltema.se/bil—mc/bilreservdelar/torkarutrustning/spolarpumpar/vindrutespolarpump-universal-2000017965>, hämtad: 15-03-2020
- [6] *Vindrutespolarpump, universal*. Sverige. u.d. [Online]. Tillgänglig: <https://www.biltema.se/bil—mc/bilreservdelar/torkarutrustning/spolarpumpar/> hämtad: 27-05-2020
- [7] Dejan, *How I2C Communication Works and How To Use It With Arduino*, [Online]. Tillgänglig: <https://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/>, hämtad: 10-04-2020
- [8] *Definition, Markfuktighet på vegetationsytan*, 2018. [Online]. Tillgänglig: <https://web.archive.org/web/20181007085744/http://www-markinfo.slu.se/sve/>

mark/fukt/skfukt1.html,
hämtad: 9-04-2020

[9] *Tutorial, Using Capacitive Soil Moisture Sensors on the Raspberry Pi*, Switchdoc, 2018. [Online]. Tillgänglig:
<https://www.switchdoc.com/2018/11/tutorial-capacitive-moisture-sensor-grove/>,
hämtad: 9-04-2020

[10] *SEN-13637*, USA: Digikey, u.d. [Online]. Tillgänglig:
[https://www.digikey.se/product-detail/en/sparkfun-electronics/SEN-13637/1568-1670-ND/7400839?utm_adgroup=Evaluation%20Boards%20-%20Sensors&utm_source=google&utm_medium=cpc&utm_campaign=Google%20Shopping_Development%20Boards%2C%20Kits%2C%20Programmers&utm_term=&productid=7400839gclid=Cj0KCQjwsYb0BRCOARIsAHbLPhG-c9lanjs8wAdf-vOuqm6TQSAYcaGl3O3sSqGq44Z2a0T0zEaw7nMaAtCUEALw_wcB\)\)](https://www.digikey.se/product-detail/en/sparkfun-electronics/SEN-13637/1568-1670-ND/7400839?utm_adgroup=Evaluation%20Boards%20-%20Sensors&utm_source=google&utm_medium=cpc&utm_campaign=Google%20Shopping_Development%20Boards%2C%20Kits%2C%20Programmers&utm_term=&productid=7400839gclid=Cj0KCQjwsYb0BRCOARIsAHbLPhG-c9lanjs8wAdf-vOuqm6TQSAYcaGl3O3sSqGq44Z2a0T0zEaw7nMaAtCUEALw_wcB))),
hämtad: 15-04-2020.

[11] ”Temperaturgivare”, i *Nationalencyklopedin*. [Online]. Tillgänglig: <https://www.ne.se/uppslagsverk/>
hämtad: 24-04-2020

[12] *Resistorer och potentiometrar*, Sverige: Kjell & Company, 2017. [Online]. Tillgänglig:
<https://www.kjell.com/se/kunskap/hur-funkar-det/arduino/grundlaggande-elektronik/resistorer-och-potentiometrar>
hämtad: 23-02-2020

[13] K. O Jeppson, ”Transistor”, i *Nationalencyklopedin*. [Online]. Tillgänglig: <https://www.ne.se/uppsla>
hämtad: 31-03-2020

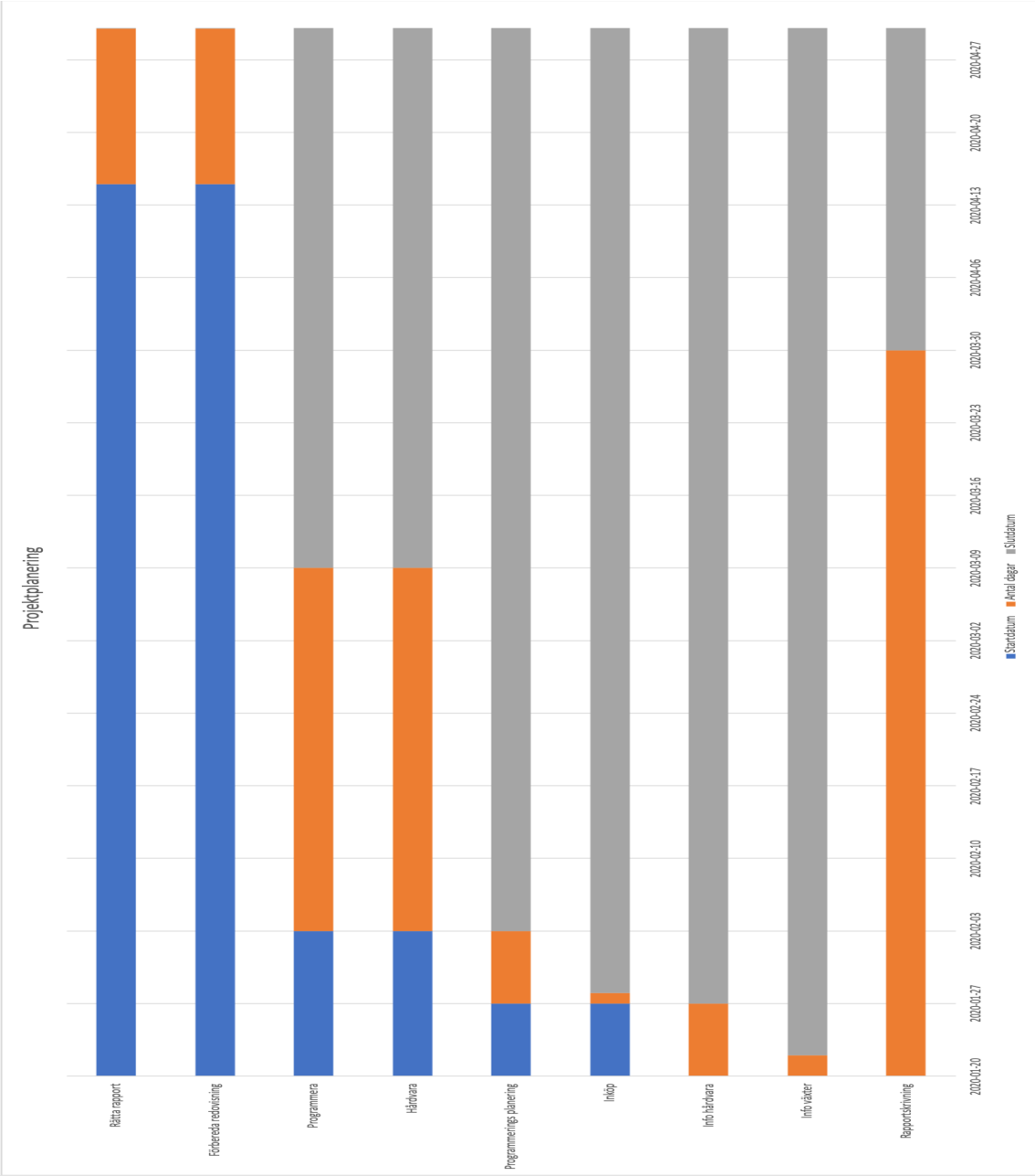
[14] L. Nordlund och I. Wiklund, *Grundläggande elektronik*. 2 uppl. Sverige: Liber AB, 2012.

[15] *101020614 - Grove kapacitiv fuktsensor, Seeed Studio*, Sverige u.d [Online]. Tillgänglig:
<https://www.elfa.se/sv/grove-capacitive-moisture-sensor-seeed-studio-101020614/p/30135124>,
hämtad: 26-02-2020

- [16] *Nivåbrytare NC*, Sverige, u.d. [Online]. Tillgänglig: <https://www.kjell.com/se/produkter/el-verktyg/elektronik/elektromekanik/strombrytare/nivabrytare/nivastrombrytare-nc-p36037>
hämtad: 27-05-2020
- [17] *X-koppling 5mm*, Nykvarn, Sverige: do88, u.d. [Online]. Tillgänglig: <https://www.do88.se/sv/artiklar/x-koppling-5mm.html>,
hämtad: 25-03-2020
- [18] *Reducering 6,3-8mm*, Nykvarn, Sverige: do88, u.d. [Online]. Tillgänglig: <https://www.do88.se/sv/artiklar/reducering-63-8mm.html>,
hämtad: 25-03-2020
- [19] *Luxorparts Transistor TIP120 4-pack*, Sverige: Kjell & Company, u.d. [Online]. Tillgänglig: <https://www.kjell.com/se/produkter/el-verktyg/elektronik/halvledare/transistorer/luxorparts-transistor-tip120-4-pack-p90752>
hämtad: 05-03-2020

Bilagor

Bilaga 1: Gantt-schema

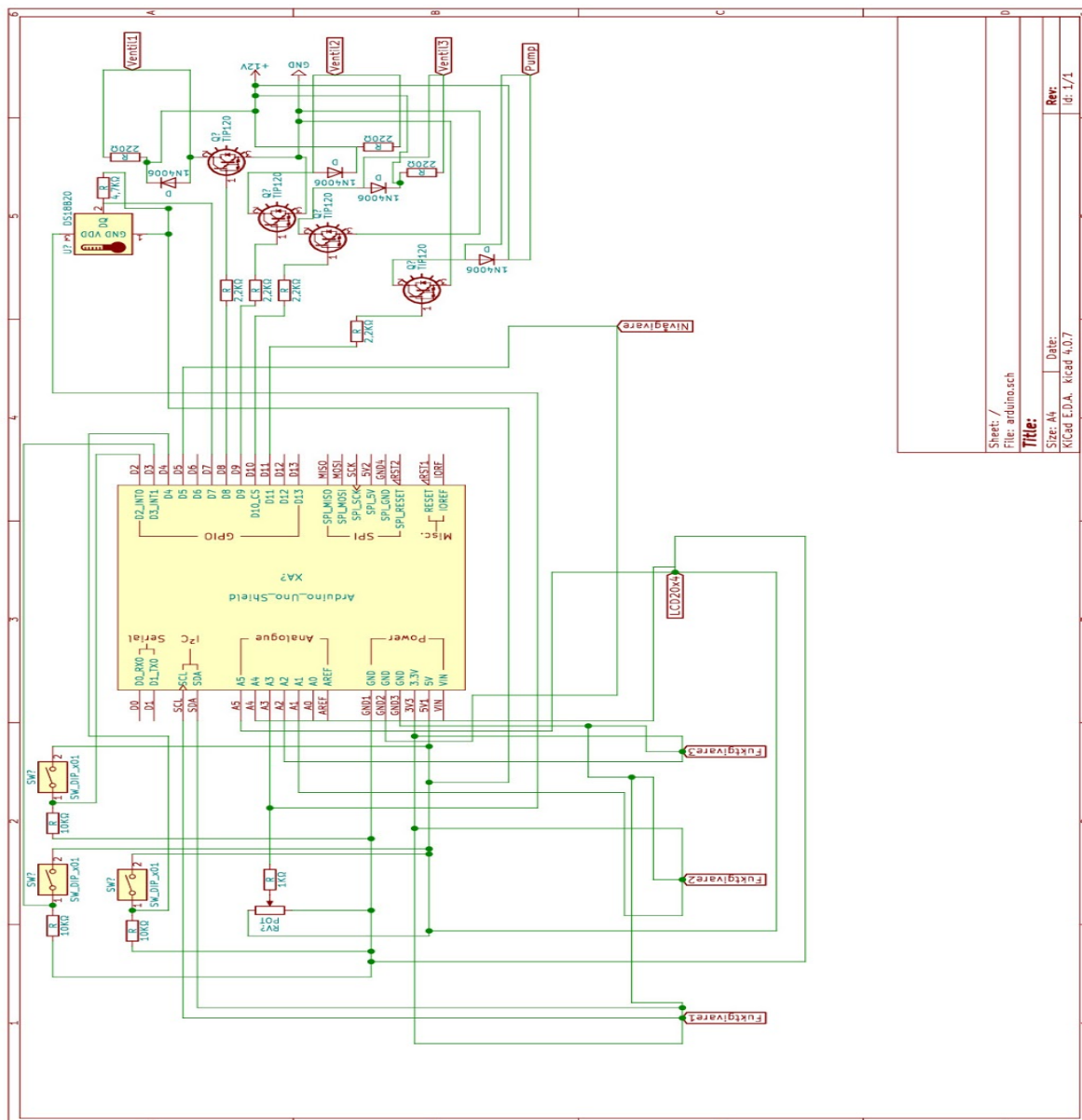


Bilaga 2: Programkod för hur potentiometer ändrar värden till procent mellan 0 och 100

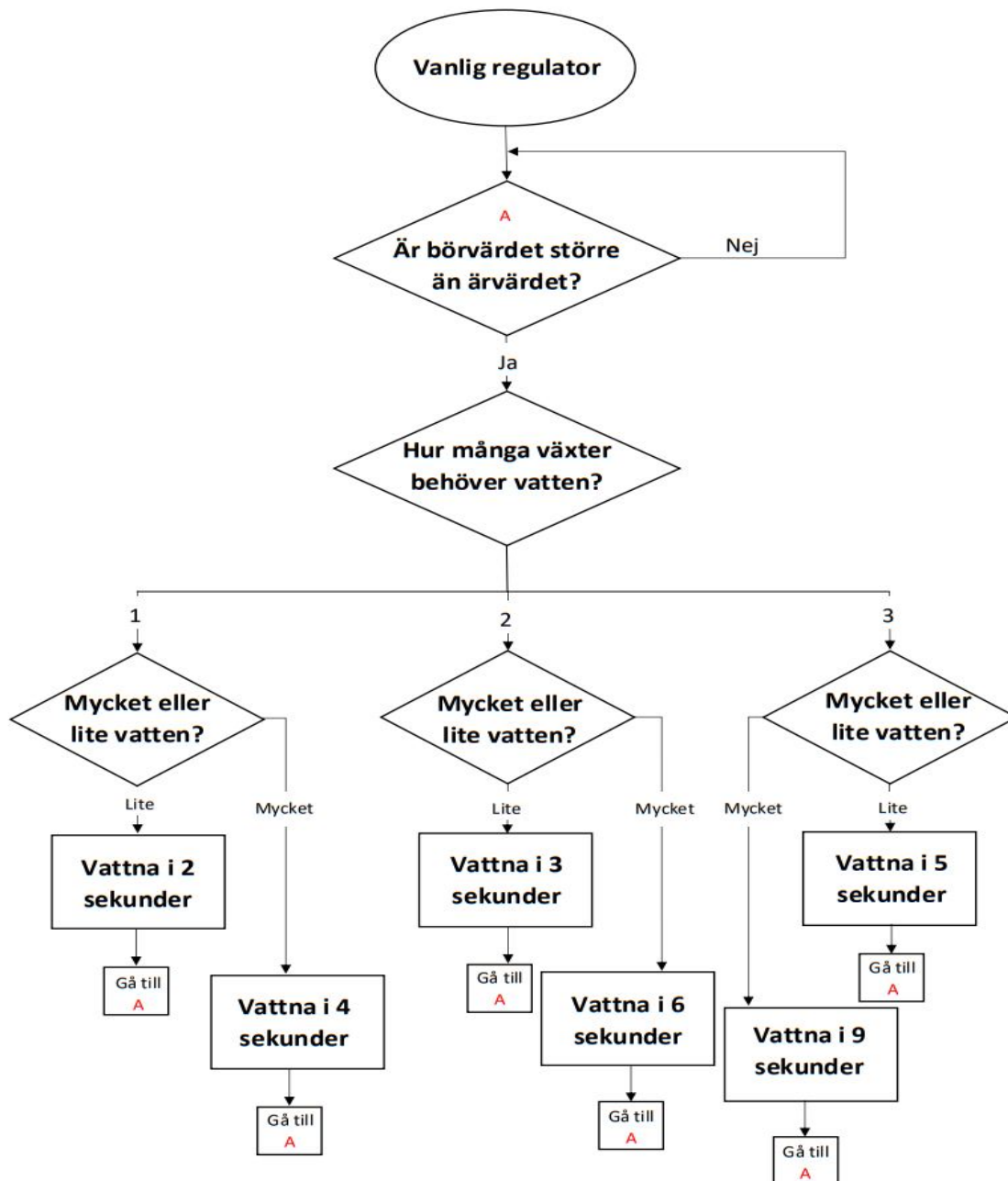
```
638 int potentiometer1(){
639   if(millis() >= (pot1 + 500)){
640     vardepot=analogRead(A3);
641     borvarde1=map(vardepot, 1023, 0, 0, 100);    //Gör om värdena så de bli 0-100
642     lcd.setCursor(10, 0);
643     lcd.print(borvarde1);
644     if (borvarde1 < 100){
645       lcd.setCursor(12, 0);
646       lcd.print(" ");
647     }
648     pot1 += 1000; }
649     return(borvarde1); }
---
```

// Så att den inte blinkar massor
//Returnerar satta börvärdet för slang 1

Bilaga 3: KiCad för kopplingsschema



Bilaga 4: Flödesschema för vanliga regulatorn



Bilaga 5: Flödesschema för torkningsregulatorn

