Automatic Waterpump



|  |  |
| --- | --- |
| **Namn** | Amin Nazari |
| **Utbildning** | Mjukvaruutvecklare inom inbyggda system |
| **Uppdrags-givare** | Cervitrol AB, Fielder & Lundgren |
| **Handledare/ examinator** | Tony Månsson |
| **Datum** | 28/04-2025 |

Sammanfattning

Detta examensarbete handlar om utveckling av ett automatiskt bevattningssystem baserat på mikrokontrollern ESP32 nano, med komponenter som jordfuktighets- och vätskenivåsensor, MQTT-kommunikation och en LCD-display och en knapp. Projektets syfte var att skapa ett smart vattensystem som självständigt kan detektera behovet av att vattna och efter det aktivera en vattenpump vid behov, samt möjliggöra fjärrstyrning och statusövervakning i realtid.

Systemet bygger på två skript. Det första skriptet ansvarar för automatisk aktiveringen av pumpen baserat på sensordata. Om jorden är torr och det finns vatten i behållaren, startas pumpen automatiskt.

Det andra skriptet hanterar LCD-displayen och en knapp som manuellt kan aktivera pumpen. All kommunikation sker via MQTT-protokollet med en publik broker där statusmeddelanden skickas och tas emot. LCD-displayen visar den aktuella statusen, och användaren kan också påverka systemet via MQTT-meddelanden, till exempel för att manuellt slå på pumpen. En knapp används för att skicka dessa kommandon, det ökar användarvänligheten.

Innehållsförteckning

[Inledning 1](#_Toc195533308)

[Bakgrund 1](#_Toc195533309)

[Syfte 1](#_Toc195533310)

[Problemformulering 2](#_Toc195533311)

[Avgränsningar och fokus 2](#_Toc195533312)

[Metod/Arbetssätt 3](#_Toc195533313)

[Teknik 4](#_Toc195533314)

[ESP32 nano – systemets hjärna 4](#_Toc195533315)

[Jordfuktighetssensor – känner av torr jord 4](#_Toc195533316)

[Vattennivåsensor – ser om det finns vatten 4](#_Toc195533317)

[Relä – slår på och av pumpen 4](#_Toc195533318)

[Vattenpump – vattna blommorna 5](#_Toc195533319)

[LCD-display – visar status 5](#_Toc195533320)

[Knapp – för manuell styrning 5](#_Toc195533321)

[Programmering med MycroPython och Thonny 5](#_Toc195533322)

[Referenslista 6](#_Toc195533323)

# **Inledning**

## Bakgrund

Under utbildningen till mjukvaruutvecklare för inbyggda system har vi fått arbeta med olika tekniker för att kombinera hårdvara och mjukvara. För att fördjupa mina kunskaper valde jag att utveckla ett automatiskt bevattningssystem som examensarbete.

Jag har själv ett behov av denna typ av produkt, eftersom jag reser mycket och inte alltid kan vara hemma för att ta hand om mina växter. Tidigare har jag bett vänner att vattna dem, men det har flera gånger lett till att växterna över vattnats och dött. Jag ville undvika detta och behövde ett smart bevattningssystem som kan känna av när jorden är torr och automatiskt tillföra vatten.

Arbetet gav mig också chansen att använda det jag har lärt mig om inbyggda system i ett praktiskt projekt som jag själv har nytta av.

## Syfte

Syftet med det här examensarbetet är att skapa ett automatiskt bevattningssystem som kan känna av om jorden är torr eller fuktig och om det finns vatten i en behållare. När det behövs ska systemet själv kunna starta en pump och vattna. Arbetet bygger på mikrokontroller ESP32 för att styra och följa systemet på distans.

Arbetet syfte ska ge en bättre förståelse för hur man utvecklar inbyggda system där hårdvara, mjukvara och nätverkskommunikation sammankopplas i ett praktiskt projekt. Och framför allt få mina blommor att må bra.

Följande funktioner som systemet ska uppfylla:

* En pump ska automatiskt aktiveras när jorden är torr och det finns vatten i behållaren.
* En LCD-display ska visa aktuell status för systemet.
* En knapp ska kunna användas för manuell aktivering av pumpen.
* En MQTT-anslutning ska möjliggöra fjärrövervakning och fjärrstyrning av systemet.

## Problemformulering

För att bygga ett automatiskt bevattningssystem som verkligen fungerar så finns det flera saker jag måste lösa. Systemet behöver kunna känna av rätt information, ta rätt beslut efter informationen och fungera säkert.

* Hur kan användaren övervaka och styra systemet via MQTT och en knapp?
* Hur kan ett system automatiskt avgöra när växter behöver vattnas?
* Hur kan man kombinera olika data för att göra smarta beslut i det automatiska systemet.

## Avgränsningar och fokus

I detta arbete ville jag fokusera på att bygga ett enkelt automatiskt bevattningssystem som använder sensorer, mikrokontroller, knapp, LCD-display, pump och MQTT-kommunikation. Målet har varit att skapa ett system som fungerar i praktiken och som jag själv har nytta av.

Jag har inte valt att:

* Utveckla en mobilapp eller webbgränssnitt
* Använda andra sensortyper, t.ex temp – eller ljussensor
* Bygga ett eget MQTT-broker system (använde en publik broker i stället)
* Designa en fysisk låda till produkten

Jag har valt att fokusera på:

* Få sensor, pump och MQTT att fungera ihop.
* Visa status på en LCD-display.
* Testa manuell styrning med en knapp.
* Bygga ett stabilt och enkelt system som är lätt att använda.

## Metod/Arbetssätt

Arbetet började med att jag funderade över vilka komponenter som skulle behövas för att bygga ett automatiskt bevattningssystem. Jag tänkte igenom vilka sensorer som behövde användas, vad de skulle känna av och vad som skulle aktiveras när vissa värden uppnåddes. Jag jämförde olika typer av sensorer och valde ut de som passade bäst för mitt arbete. Därefter gjorde jag en lista över vilka komponenter som behövdes. Jag valde att använda en jordfuktighetssensor för att mäta fuktnivån i jorden och en vattennivåsensor för att kontrollera om det fanns vatten i behållaren. Från början var det inte min tanke att ha med en knapp för manuell styrning eller en sensor som läser av vattennivå, men under arbetets gång insåg jag att det skulle göra systemet båda användarvänligt och stabilare.

Innan jag byggde ihop hela produkten testade jag varje komponent var för sig. Det hjälpte mig att förstå hur de fungerade och att upptäcka möjliga problem tidigt. Jag använde databladen för att förstå hur varje komponent skulle kopplas och programmeras. För att skriva och testa koden använde jag Thonny som utvecklingsmiljö eftersom det fungerar bra med ESP32 Nano när man programmerar i Mycropython.

MQTT-kommunikationen var ett område vi har lärt oss om på flera lektioner, där vi har gått igenom hur enheter kan kommunicera med varandra trådlöst. Det blev ett enkelt val att välja just MQTT protokollet som en lösning för att kunna övervaka och styra systemet på distans. Jag använde en publik MQTT-broker för att skicka och ta emot meddelande mellan ESP32:n och min dator.

När alla komponenter fungerade var för sig, så började jag bygga upp systemet stegvis. Jag började med att koppla upp sensorerna till ESP32 NANO och skriva kod för att läsa data från dem. Jag stötte på problem när jag skulle styra vattenpumpen. Eftersom till början saknade jag ett relä som kunde styra pumpen. När jag fick tag på ett relä kunde jag därefter koppla in och styra pumpen korrekt via mikrokontroller. Jag la också till en LCD- och knapp för användaren enkelt skulle kunna styra pumpen manuellt och se aktuell status i systemet. Display visade till exempel om jorden var torr eller om vattennivån var låg.

Jag testade också olika scenarier som till exempel torr eller låg vattennivå, för att se hur systemet reagerade. Det gav mig möjlighet att justera koden och säkerställa att systemet fungerar stabilt. Under testningen märkte jag att jag behövde justera tröskelvärdet för att systemet skulle reagera mer exakt när jorden blev våt och torr.

# **Teknik**

Här är vad för teknik som användes i arbetet. Jag går igenom vilken mikrokontroller jag valde, vilka sensorer och andra komponenter som användes. Hur kommunikationen fungerar och vilket programmeringsverktyg jag jobba med.

## ESP32 nano – systemets hjärna

ESP32 Nano är en smart mikrokontroll som styr hela systemet. Den fungerar som hjärnan i projektet. Den har inbyggd Wi-Fi, vilket gör att man kan kopplas upp trådlöst mot min dator. ESP32 har flera in och utgångar där man kan koppla in sensorer, knappar, displayer och andra komponenter. I mitt arbete används den för att läsa jordfuktighet och vattennivå, visa information på en LCD-display, ta emot tryck från en knapp och styra en pump.

Jag valde denna mikrokontroller för den är enkel att använda och har de funktioner jag behövde. Den klarar av att både läsa data och kommunicera trådlöst med MQTT utan problem.

## Jordfuktighetssensor – känner av torr jord

Jordfuktighetssensor mäter hur mycket vatten det finns i jorden. Den fungerar genom att mäta motståndet mellan två metallpinnar. När jorden är torr så är motståndet högt, och när jorden är blöt så är motståndet lägre. Sensorn ger ett värde som mina mikrokontroller kan läsa.

## Vattennivåsensor – ser om det finns vatten

Vattennivåsensor kan känna av vätska genom plats eller andra icke metalliska behållare. Sensorn sätts utanpå en vattenbehållare och känner av om det finns vätska på insidan eller inte. Det gör att man inte behöver ha någon sensor i direkt kontakt med vatten.

## Relä – slår på och av pumpen

Reläet fungerar som en elektrisk strömbrytare. ESP32 kan inte driva pumpen direkt så används reläet för att koppla på eller av pumpens ström. Reläet styrs med en signal från mikrokontrollen och kan kopplas så att det aktiveras när jorden är torr.

## Vattenpump – vattna blommorna

Pumpen som används är en liten nedsänkbar pump. Den drivs med låg spänning men har tillräckligt tryck för att vattna en blomkruka. Pumpen är kopplad till reläet som styr när den ska sättas igång.

## LCD-display – visar status

LCD-displayen som används är en 16x2. Det innebär att den kan visa 16 tecken och på 2 rader. Den visar aktuella statusen i systemet. Som till exempel om jorden är torr, om det finns vatten i behållaren eller om pumpen är igång.

Displayen är kopplad till mikrokontrollen med hjälp av I2C-protokollet, vilket gör att man bara behöver två signalkablar (SDA och SCL) i stället för många. Det förenklar kopplingen och sparar plats på breadboarden.

## Knapp – för manuell styrning

Knappen används om användaren vill starta pumpen manuellt. När knappen trycks ner skickas en signal till mikrokontroller som då aktiverar pumpen med en fras ”turn\_on”.

## Programmering med MycroPython och Thonny

Mikrokontrollen är programmerad med MycroPython. Jag använde Thonny som utvecklingsmiljö för att skriva och testa koden. Det fungerar bra för att ladda upp koden och se resultat direkt på mikrokontrollen.

# **Referenslista**

Fuktighetssensor:

<../../../Downloads/jordfuktighetsensor.pdf> - Datablad

<https://www.electrokit.com/jordfuktighetssensor?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCQKw1AHlok85Nuw5rnUQ3h7kRUEIl9FhM-hfB2ceNa8j9I6ocfwq1saAiRqEALw_wcB>

Vattennivåsensor:

<../../../Downloads/Vattennivåsensor.pdf> - Datablad

<https://www.electrokit.com/vatskenivasensor-kontaktlos-kapacitiv-5-24vdc?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCRjOqXW9nlYKh_EWpIQJBIJc3ao94Ur7akfAMDGN5bXcdcUMiBcJZEaAoD4EALw_wcB>

Dränkbar pump:

[../../../Downloads/Dränkbar pump.pdf](../../../Downloads/Dränkbar%20pump.pdf) - Datablad

<https://www.electrokit.com/drankbar-pump-3v?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCR9j-73lxJ3jd-hIS36wYmGFHdjf95iWMvCG0lLwbST41clelpcsbAaAgaVEALw_wcB>

LCD-display:

<https://www.electrokit.com/lcd-2x16-tecken-jhd162a-stn-bla/vit-led?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCQcwKA9eJLaSEcr33QKngNEJU1D7o7UhMaYdt5pqoaRc6GXrHi7RpUaAu3LEALw_wcB>

Knapp:

<../../../Downloads/knapp.jpg> - Datablad

<https://www.electrokit.com/knappar-pcb-sortiment-12st?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCRvCW6nVoX65InoMmo9orzfUvu6y3TW4IkxcfOaH2HL3pCwTam6Lf0aApV9EALw_wcB>

Relä:

<../../../Downloads/relä.pdf> - Datablad

<https://www.electrokit.com/relamodul-5v?gad_source=1&gbraid=0AAAAAD_OrGPwO1eEjW7exDbQY0-OVBiYt&gclid=Cj0KCQjwzYLABhD4ARIsALySuCSrjVkKk-YsQTnMZBjNnS2ezeUCHttwKidIqt4wuif6qPRHtcUXz4saArM4EALw_wcB>