Tampereen ammattikorkeakoulu

Tutka oviavauksella

Sulautetut järjestelmän harjoitustyö

Team ENER-Gy

Eino Lausmaa
Eemeli Ranta
Nhan Phan

SISÄLLYS

1 JOHDANTO

Tämän projektin tavoitteena on luoda ryhmän kanssa pieni sulautettuihin järjestelmiin liittyvä mikropiirirakennelma. Rakennelman oli koostuttava useammasta komponentista, joista keskeisimpänä on Arduino Nano –mikrokontrolleri. Kontrollerin eri ominaisuuksia hyödyntäen projektissa rakennetaan pieni elektroninen järjestelmä, jonka aihe on vapaasti valittavissa ryhmän kesken.

Aiheena ryhmä valitsi ovenavausjärjestelmän kääntyvällä tutkalla. Tämän järjestelmän rakentaminen oli sopiva valinta kurssin tavoitteisiin nähden, sillä sen rakentamisessa on mahdollista hyödyntää monia kurssilla opittuja asioita, kuten servoja tutkalle ja ovelle, ultraäänianturia etäisyyden havaitsemiseen, keskeytyksiä koodin kysymiseen ja jopa rekisteriä servon ajoituksen hallintaan.

2 TAVOITTEET

2.1 Mihin projektissa pyrittiin

Projektissa tavoitteena oli suhteellisen yksinkertaiseen toteutukseen sisällyttää monia ominaisuuksia, jotka liittyivät kurssilla läpikäytyihin aiheisiin. Näillä ominaisuuksilla pyrittäisiin arvosanaan 5, jonka vaatimukset ovat:

- *Työssä on käytetty jotain omaa ja innovatiivista ratkaisua.*
- *Työssä on käytetty neljää sisäistä IO-lohkoa, joista vähintään kahta on käytetty suoraan rekisteriohjauksen kautta.*
- *Työssä on käytetty kahta erilaista keskeytyspalvelua.*
- *Laitteella on jokin tiedonsiirtoyhteys (serial monitoria ei hyväksytä)*
- *Työ on kokonaisuudessaan erittäin monipuolisesti ja tarkasti tehty ja siihen on selkeästi panostettu!*

2.2 Kuinka näitä tavoitteita lähettiin lähestymään

Oven avaus tutkan avulla saattaa kuulostaa aluksi yksinkertaiselta, mutta siihen saisi lisättyä yllättävän paljon erilaisia lisäominaisuuksia. Näitä lisäominaisuuksia voisi esimerkiksi olla vaatimuksiin liittyen:

Innovatiivinen ratkaisu	Käyttää jotain uutta kirjastoa (Esimerkiksi Software Serial -tiedonsiirtoyhteyttä varten)
Neljä sisäistä IO-lohkoa	D, vahtikoiraa voisi käyttää järjestelmävikojen huomaamiseen, ja EEPROM –muistia tallentamaan esim. Oven avauksien määrää, tai turvakoodia, I2C voisi käyttää näytön yhdistämiseen, ja etäsyyttä mittaamalla analogisignaalista digitaaliseksi ADC –muuntimella.
Kaksi keskeytystä	Keskeytykset voisi tehdä, kun ovelle lähestytään, ja
Tiedonsiirtoyhteys	Kaksi Arduino Nanoa keskustelisivat keskenään, jakamalla kuormitusta laitteiden käytöstä
Panostus	Rakennetta voisi ehostaa 3D –tulostetuilla osilla

Näiden ominaisuuksien tarkempiin toteutuksiin paneudutaan myöhemmin.

3 SUUNNITTELU

3.1 Työnjako

Tässä projektissa haastetta toi se, että Eemeli jäi osaksi aikaa pois projektin tekemisestä (n.2 viikkoa) Joten hänelle isommaksi työosuudeksi jäi dokumenttien täyttö. Nhan keskittyy koodin rakentamiseen, ja Eino toimi hardisvastaavana. Tietäen myös osallisen työvajauden projektin aikana, projektin toteutus pidettiin maltillisena, että se ehdittäisiin saamaan paremmin aikamääreissään valmiiksi.

3.2 Ominaisuudet

Rakenteeseen tarvittaisiin mukaan vähintäänkin yksi etäisyysanturi, ja servomoottori, jolla tunnistetaan lähelle saapuva henkilö, sekä voitaisiin avata ovi. Jotta kuka tahansa ei pääsisi ovesta läpi, siihen tulee mukaan vielä näppäimistö salasanalla, ja näyttö, josta näkee eri oven tilat salasanan kirjoituksen aikana. Näppäimistön kirjoituksen hoitaa toinen Nano, jonka avulla saadaan kuormaa ja pinnipaikkoja vapautettua ykkösnanolta. Erillisenä ilmoittimena piiriin tulee kaksi eriväristä LED-diodia (vihreä ja punainen) jotka osoittavat koodin antamisen tilan. Jotta tutka näkisi paremmin ympärilleen, sen alle asennetaan toinen servomoottori, joka kääntelee etäisyysanturia n.180 asteen kulmassa. Äänimerkin antava kaiutin/hälytin ilmoittaa vielä tilan erikseen valon ja näytön lisäksi.

3.3 Komponenttien valinta

Ominaisuuksien perusteella alle on kerätty alustava komponenttilista:

- 2kpl Arduino Nano
- Ultraäänianturi
- 2kpl Servomoottoreita
- LCD-näyttö
- 4x4 näppäimistö
- Kaiutin/hälytin äänimerkille

- 2kpl LED-diodeja, punainen ja vihreä
- Kondensaattori
- 3kpl vastuksia

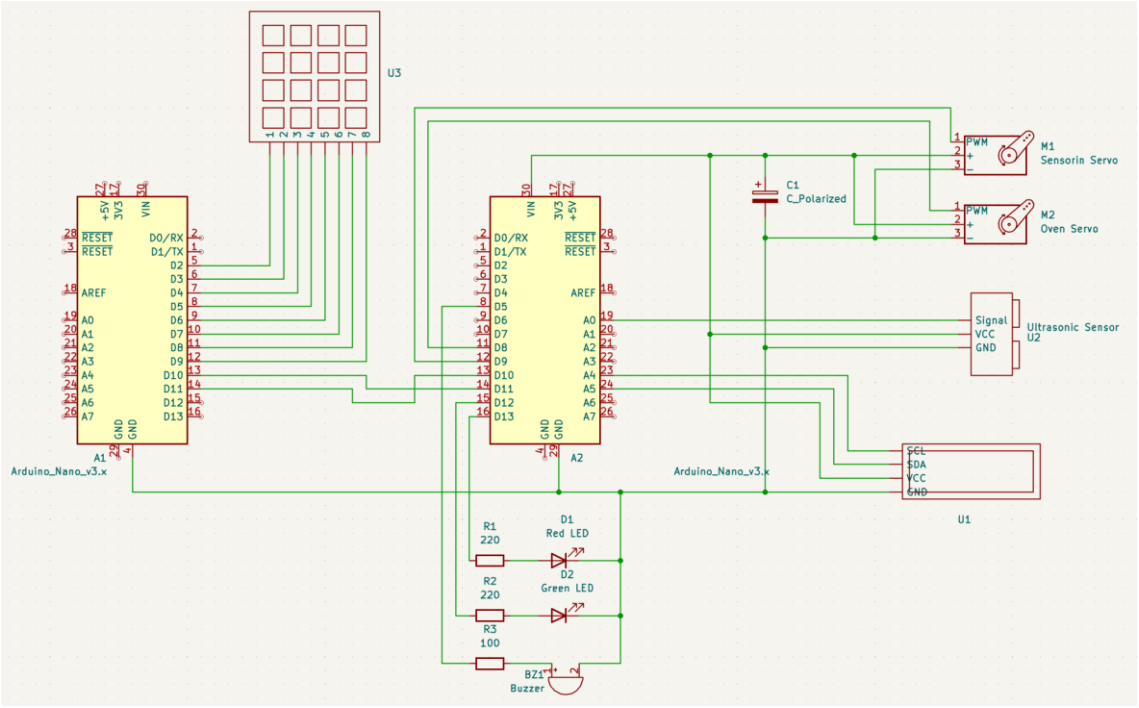
4 TOTEUTUS

4.1 Piiri

Kaikki komponentit eivät mahtuisi yhdelle pitkällekään näkkileivälle, joten piriin on lisättävä ainakin yksi osittainen näkkileivän osa, jotta kaikki saadaan mahtumaan mukaan. Toteutuksessa päädyimme laittamaan näppäimistön eri alustalle, koska se on isoin yksittäinen komponentti projektissa. Arduinot asetettiin vastakkain isomman näkkileivän eri päihin, ja pienemmät elektroniikkakomponentit tulevat näiden väliin keskelle.

#2 Nano toimii näppäimistön syötteen vastaanottimena, ja lähettää tiedon #1 Nanolle sarjayhteyttä pitkin, kun #1 Nano ohjaa taas kaikkia muita toimintoja. Näiden välinen yhteys on toteutettu ohjelmistopuolen sarjaväylällä käyttäen kirjastoa SoftwareSerial.h, koska normaalia sarjaväyläliitosta käyttäessä syntyi ongelmia koodin lataamisessa ja käynnistyksissä.

Näyttönä toimii DFRobotin LCD 1602 näyttö, jota ohjataan I2C-tiedonsiirtoväylän kautta A4(SDA) ja A5(SCL) pinneistä. Tämä säästää pinnejä, kun tarvitaan vain 2 tiedonsiirtopinniä näytön käyttöön. Ultraäänianturi URM09 samalta valmistajalta toimii etäisyysanturina tutkan roolissa, jonka dataa lähetetään A0 pinnille luettavaksi. Servojen ohjaus tapahtuu pinneistä D8 ja D9, joita ohjataan pulssinleveysmodulaatiolla, säätämällä kanttiaallon pulssin leveyttä, jota lähetetään servon PWM pinniin. LED-diodien ohjaus suoritetaan D12 ja D13 pinneistä vastuskytkennöillä, sekä äänimerkkilaite (kaiutin) on kytkettynä D5 pinniin äänimerkkiä varten vastuksen kanssa.



Kuva 1 Piirikaavio

4.2 Muut rakennelmat

Muiden rakenteiden (ovi, tutkan alusta) materiaalina käytimme 3D-tulostusta projektin mukana tulleen puualustan lisäksi. Tutkaa varten teimme ultraäänianturille telineen, jolla sen voi kiinnittää servoon kiinni. Itse ovi ja sen avausmekanismi on myös tulostettu.

Itse tulostuksen kanssa tuli kylläkin muutamia haasteita. Tulostimen käynnistäminen vei aikaa, koska nettiyhteys tulostimen kanssa oli epävakaa. Toisena haasteena, koska vain yksi tulostin oli käytössä, sen käyttöä piti jakaa muiden kanssa, joten se ei aina ollut saatavilla. Oven kanssa myös ongelmaksi muodostui tulostimen yhtäkkinen pysäytys tulostukselle. Ovi ja sen pylväät jäivät vajaiksi, koska tulostus ei tulostanut koko ovea loppuun, vaan jätti sen osittain tulostamatta, ja siltikin väitti tulostuksen olevan valmis. Tätä varten rakennelmaa joutui hieman improvisoimaan erilaisilla kiinnityksillä.

4.3 Koodi

Alle on lisätty kommentoidut koodit kummastakin Nanosta.

Nano #1

```
#include <Servo.h>
#include "DFRobot_RGBLCD1602.h"
#include <SoftwareSerial.h>
#include <EEPROM.h>
#include <avr/wdt.h>
SoftwareSerial link(10,11);

// --- Password / UI state ---
bool passwordRequired = false;
String password = "2025";
String inputCode = "";

// --- LEDs ---
const int green = 12;
const int red = 13;

// --- LCD setup ---
const int colorR = 255;
const int colorG = 255;
const int colorB = 255;
DFRobot_RGBLCD1602 lcd(/*RGBAddr*/0x60 ,/*lcdCols*/16,/*lcdRows*/2);

// --- Servo & Sensor ---
Servo myServo;
Servo doorServo;
const int CLOSED_ANGLE = -2;
const int OPEN_ANGLE = 60;

#define SERVO_PIN 9
#define SERVOD_PIN 8
#define SENSOR_PIN A0 // Analog sensor pin
#define MAX_RANG 520 // cm
#define ADC_SOLUTION 1023.0 // 10-bit ADC

// --- Buzzer/Speaker ---
#define BUZZER_PIN 5
int melody[] = {262, 294, 330, 349, 392, 440, 494, 523}; // Do-Re-Mi-Fa-So-La-Si-Do
int noteDuration = 400; // milliseconds

// --- Watchdog ---
#define WATCHDOG_TIMEOUT WDTO_8S



// --- LOCK-KEYPAD ---
int wrongAttempts = 0;
bool keypadLocked = false;
unsigned long lockoutUntil = 0;
const unsigned long LOCKOUT_MS = 60000; // 1 minute
```

```
// --- Servo sweep via Timer2 ISR ---
volatile int angle = 0;
volatile int step = 2; // degrees per tick
volatile byte tickCount = 0;
unsigned long passwordStartTime = 0;


// --- Helper functions ---
float getDistance() {
float sensorValue = analogRead(SENSOR_PIN);
float distance = sensorValue * MAX_RANG / ADC_SOLUTION;
if (distance > MAX_RANG) distance = MAX_RANG;
return distance;
}

void safeDelay(unsigned long ms) {
unsigned long start = millis();
while (millis() - start < ms) {
wdt_reset();
delay(10);
}
}


// --- Melody helper ---

void playTone(int freq, int duration) {
long period = 1000000L / freq;
long cycles = (long)duration * 1000L / period;
for (long i = 0; i < cycles; i++) {
digitalWrite(BUZZER_PIN, HIGH);
delayMicroseconds(period / 2);
digitalWrite(BUZZER_PIN, LOW);
delayMicroseconds(period / 2);
}
}

void playMelody() {
for (int i = 0; i < 8; i++) {
wdt_reset();
playTone(melody[i], noteDuration);
delay(50);
}
}

void playSuccessMelody() {
int notes[] = {1046, 1318, 1567, 2093, 1567};
int count = sizeof(notes) / sizeof(notes[0]);

for (int i = 0; i < count; i++) {
playTone(notes[i], 180);
delay(20);
}
}
```

```
void playlockWarningMelody() {
// Rising alarm sequence
playTone(500, 150);
delay(180);
playTone(600, 150);
delay(180);
playTone(700, 150);
delay(180);
playTone(900, 200);
delay(250);

// Falling melodic tones
playTone(750, 180);
delay(200);
playTone(620, 180);
delay(200);
playTone(480, 250);
delay(300);

// Deep final lockout tone
playTone(300, 600);
delay(650);
}

void playTimeoutMelody() {
// Descending tones
playTone(800, 150); // First tone
delay(100);
playTone(600, 150); // Second tone
delay(100);
playTone(400, 250); // Final tone
delay(200);
}



// --- EEPROM helpers ---
void loadPassword() {
char stored[10] = {0};
for (int i = 0; i < 10; i++) {
char c = EEPROM.read(i);
if (c == '\0' || c == 0xFF) break; // Stop at null or empty
stored[i] = c;
}
password = String(stored);
if (password.length() == 0) password = "2025"; // Default
}

void savePassword(String newPass) {
for (int i = 0; i < newPass.length(); i++) {
EEPROM.write(i, newPass[i]);
}
EEPROM.write(newPass.length(), '\0');
}



void openDoorSequenceBlocking() {
// Open door
doorServo.write(OPEN_ANGLE);
unsigned long start = millis();
```

```
safeDelay(500);
lcd.clear();
lcd.print("Door Ready");
// Wait 10 seconds
safeDelay(10000);
lcd.clear();
lcd.print("Door Close");
// Reattach (if detached) and close
if (!doorServo.attached()) {
doorServo.attach(SERVOD_PIN);
}
doorServo.write(CLOSED_ANGLE);
delay(500);
}

// --- Process one key character from Nano #2 ---
void processKeyChar(char key) {
if(keypadLocked) return;
if (key == '#') {
// Confirm code
inputCode.trim();
if (inputCode == password) {
keypadLocked = false;
wrongAttempts = 0;
digitalWrite(red, !digitalRead(red));
digitalWrite(green, HIGH);
lcd.clear();
lcd.print("Access Granted");
lcd.setRGB(0, 255, 0);
playMelody();
openDoorSequenceBlocking();
safeDelay(3000); // UI pause
// Ask if user wants to change password
lcd.clear();
lcd.print("Change Pass?");
lcd.setCursor(0, 1);
lcd.print("Type ** or #");

String choice = "";
unsigned long startTime = millis();
while (millis() - startTime < 10000) { // 10s window
if (link.available()) {
char key = (char)link.read();
if (key == '\n' || key == '\r') continue;
choice += key;
if (choice.endsWith("**")) break; // Change password
if (choice.endsWith("#")) break; // Skip
}
}

if (choice.endsWith("**")) {
// Change password mode
lcd.clear();
lcd.print("New Code:");
String newPass = "";
while (true) {
if (link.available()) {
char k = (char)link.read();
if (k == '#') break; // Confirm new password
newPass += k;
```

```
lcd.clear();
lcd.print("New: ");
lcd.print(newPass);
}
}

if (newPass.length() > 0) {
savePassword(newPass);
password = newPass; // Update in RAM
lcd.clear();
lcd.print("Saved!");
playSuccessMelody();
safeDelay(2000);
}
} else {
wrongAttempts = 0;
keypadLocked = false;
lcd.clear();
lcd.print("No Change");
safeDelay(1000);
}
passwordRequired = false;
myServo.attach(SERVO_PIN); // resume servo motion
digitalWrite(green, LOW);
lcd.setRGB(255, 255, 255);
} else {
// Check if we reached 3 wrong attempts
if (wrongAttempts >= 2) {
keypadLocked = true;
lockoutUntil = millis() + LOCKOUT_MS;
wrongAttempts = 0;

digitalWrite(red, HIGH);
lcd.setRGB(255, 0, 0);
lcd.clear();
lcd.print("Too many wrong attempts");
lcd.setCursor(0, 1);
lcd.print("Doorpad is closed for 1 minute");

// Optional alert pattern before reset
playlockWarningMelody();
}
wrongAttempts++;
lcd.clear();
lcd.print("Wrong Code");
lcd.setCursor(0, 1);
lcd.print("Try Again!");
// Triple beep for wrong code
playTone(200, 200);
delay(100);
playTone(200, 200);
delay(100);
playTone(200, 200);
}
inputCode = "";
} else if (key == '*') {
inputCode = "";
lcd.clear();
lcd.print("Enter Code:");
} else {
```

```
// Append any other key
inputCode += key;
lcd.clear();
lcd.print("Code: ");
lcd.print(inputCode);
}
}

// --- Timer2 Compare Match ISR (non-blocking sweep) ---
// CTC mode, OCR2A chosen for ~5ms tick, accumulate to ~50ms per angle step
ISR(TIMER2_COMPA_vect) {
// Only sweep when not in password mode
if (!passwordRequired) {
tickCount++;
if (tickCount >= 10) { // 10 * ~5ms = ~50ms step
tickCount = 0;
angle += step;
if (angle >= 180 || angle <= 0) step = -step;
}
}
}



void setup() {
// WATCHDOG safety
MCUSR = 0;
wdt_disable();
wdt_enable(WDTO_8S);



// LEDs
pinMode(green, OUTPUT);
pinMode(red, OUTPUT);
pinMode(BUZZER_PIN, OUTPUT);
digitalWrite(green, LOW);
digitalWrite(red, LOW);
digitalWrite(BUZZER_PIN, LOW);

// UART (Nano-to-Nano)
Serial.begin(9600); // For debugging
link.begin(9600); // SoftwareSerial for Nano-to-Nano

// LCD
lcd.init();
lcd.setRGB(colorR, colorG, colorB);
lcd.clear();
lcd.print("ENE-gy Ovet");

// Servo
myServo.attach(SERVO_PIN);
doorServo.attach(SERVOD_PIN);
doorServo.write(CLOSED_ANGLE);



loadPassword(); // Load password from EEPROM
// savePassword("2025");
// --- Timer2 in CTC mode ---
```

```
// 16MHz / 1024 prescaler => 15625 Hz timer tick
// OCR2A=78 => (78+1)/15625 ≈ 0.005s = ~5ms
cli();
TCCR2A = 0;
TCCR2B = 0;
TCCR2A |= (1 << WGM21); // CTC mode
TCCR2B |= (1 << CS22) | (1 << CS21) | (1 << CS20);// Prescaler 1024
OCR2A = 78; // ~5ms interval
TIMSK2 |= (1 << OCIE2A); // Enable Compare A interrupt
sei();
}



void loop() {
wdt_reset(); // Normal kick → resets counter to 0
float distance = getDistance();

// If someone is close and we aren't already in password mode
if (distance < 10 && !passwordRequired) {
passwordRequired = true;
passwordStartTime = millis(); // Start timer
myServo.detach(); // stop servo motion to keep door locked
lcd.setRGB(255, 0, 0);
lcd.clear();
lcd.print("Enter Code:");
digitalWrite(red, HIGH);
playTone(400, 200); // Short beep for object detection
}

// Handle keypad input coming from Nano #2 over UART
if (passwordRequired) {
if (keypadLocked) {
// Stay locked until time is up
if (millis() >= lockoutUntil) {
keypadLocked = false;
wrongAttempts = 0;
digitalWrite(red, LOW);
lcd.setRGB(255, 255, 255);
lcd.clear();
lcd.print("Lockout over");
safeDelay(1000);
lcd.clear();
lcd.print("Enter Code:");
passwordStartTime = millis();
} else {
// Show countdown
while (millis() < lockoutUntil) {
wdt_reset();
unsigned long remaining = (lockoutUntil - millis() + 999) / 1000;
lcd.clear();
lcd.print("Keypad locked");
lcd.setCursor(0, 1);
lcd.print(remaining);
lcd.print(" s");
delay(200);
}
}
return;
```

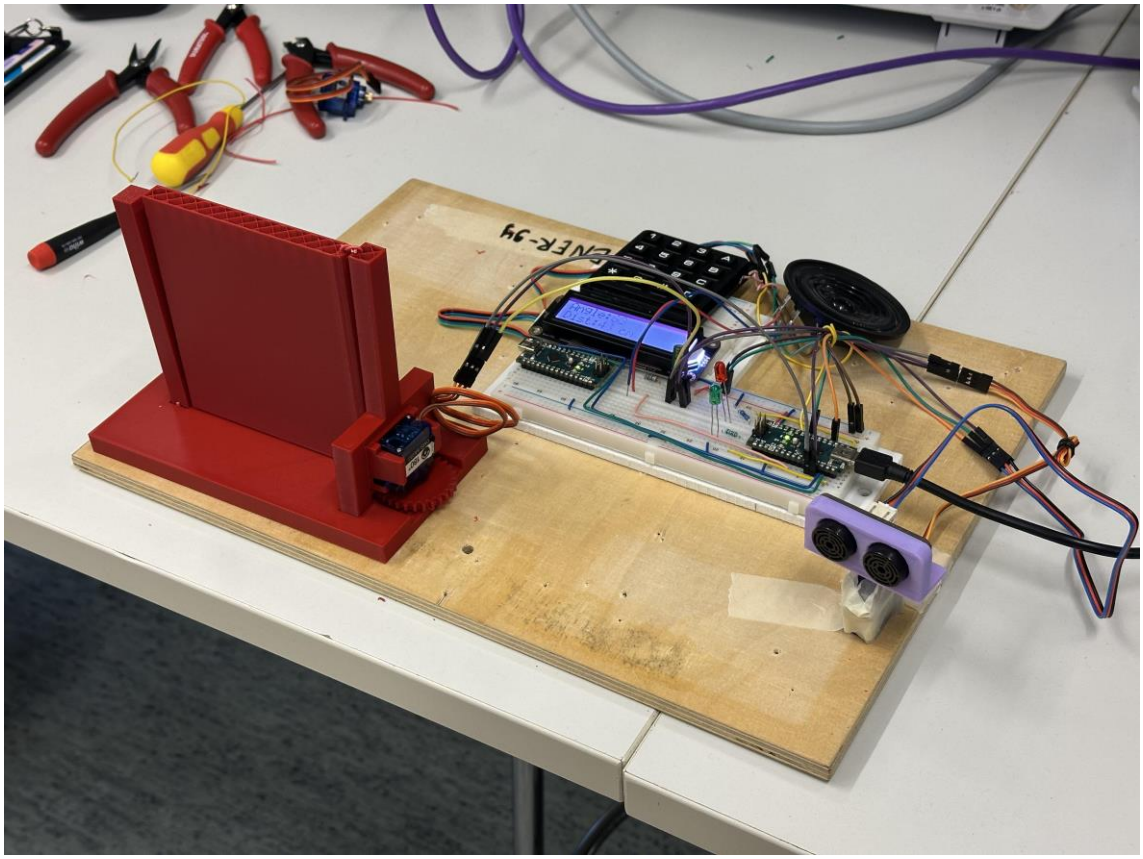
```
}

while (link.available() > 0) {
char key = (char)link.read();
if (key == '\n' || key == '\r') continue;
processKeyChar(key);
passwordStartTime = millis(); // Reset timer on input
}

if (millis() - passwordStartTime > 7000) { // 7 seconds
passwordRequired = false;
myServo.attach(SERVO_PIN); // Resume servo motion
digitalWrite(red, LOW);
lcd.setRGB(255, 255, 255);
lcd.clear();
lcd.print("Timeout!");
playTimeoutMelody();
safeDelay(2000);
keypadLocked = false;
wrongAttempts = 0;
}

while (link.available() > 0) {
char key = (char)link.read();
// Optional: skip delimiters like '\n' or '\r'
if (key == '\n' || key == '\r') continue;
processKeyChar(key);
passwordStartTime = millis(); // Reset timer on input
}
} else {
// Normal sweep display & servo update
myServo.write(angle);

// Status to Serial Monitor for debugging
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
Serial.print("Angle: ");
Serial.print(angle);
Serial.println(" astetta");

// LCD screen
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Angle:");
lcd.print(angle);
lcd.setCursor(0, 1);
lcd.print("Dist:");
lcd.print(distance, 0);
lcd.print(" cm");
}

}
```

Nano #2

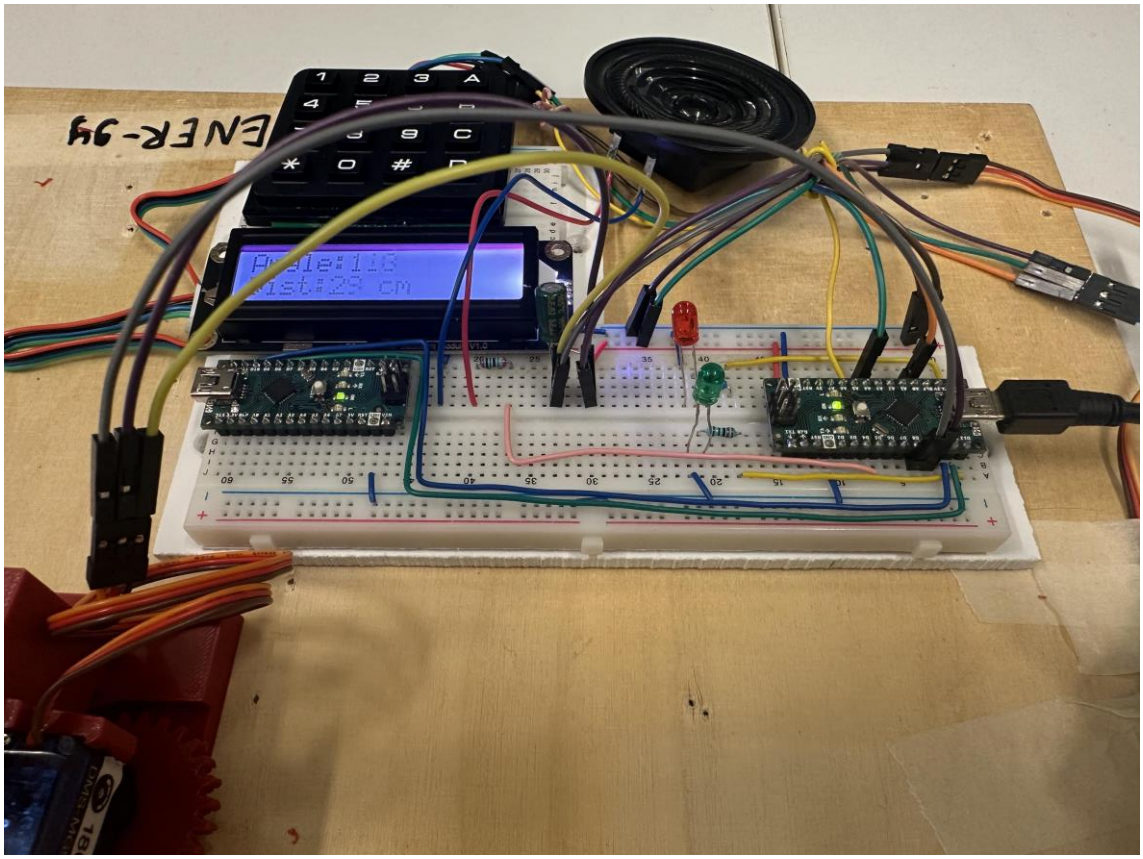
```
#include <Keypad.h>
#include <SoftwareSerial.h>

// --- Software serial setup  ---
SoftwareSerial link(10, 11);

// --- Keypad setup  ---
const byte ROWS = 4, COLS = 4;
char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};
byte rowPins[ROWS] = {5, 4, 3, 2};
byte colPins[COLS] = {9, 8, 7, 6};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
// UART for Nano-to-Nano communication using software side
link.begin(9600);
}

void loop() {
char key = keypad.getKey();
if (key) {
// Send the raw character to Nano #1
link.write(key);
}
}
```

4.4 Kokonaisuus

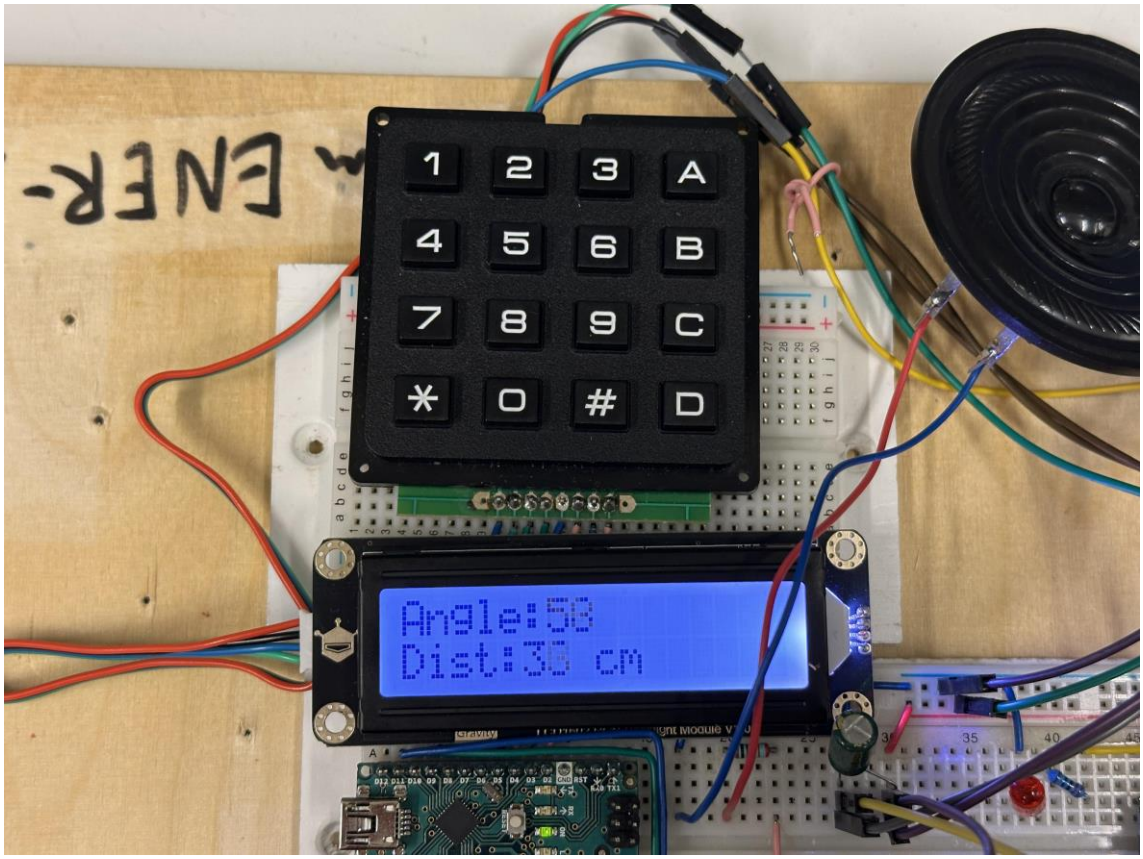
Lopputuloksena saimme toteutettua toimivan rakennelman, joka suorittaa kaikki tavoitellut toiminnot ilman ongelmia. Ulkoasu on toki vielä hyvin prototyyppimainen, koska johtoja mene hieman sekalaisesti eri suuntiin, ja kaikkia kiinnityksiä ei ehditty viimeistelemään. Ryhmä kuitenkin hyväksyi tämän lopputuloksen yhteismielin.



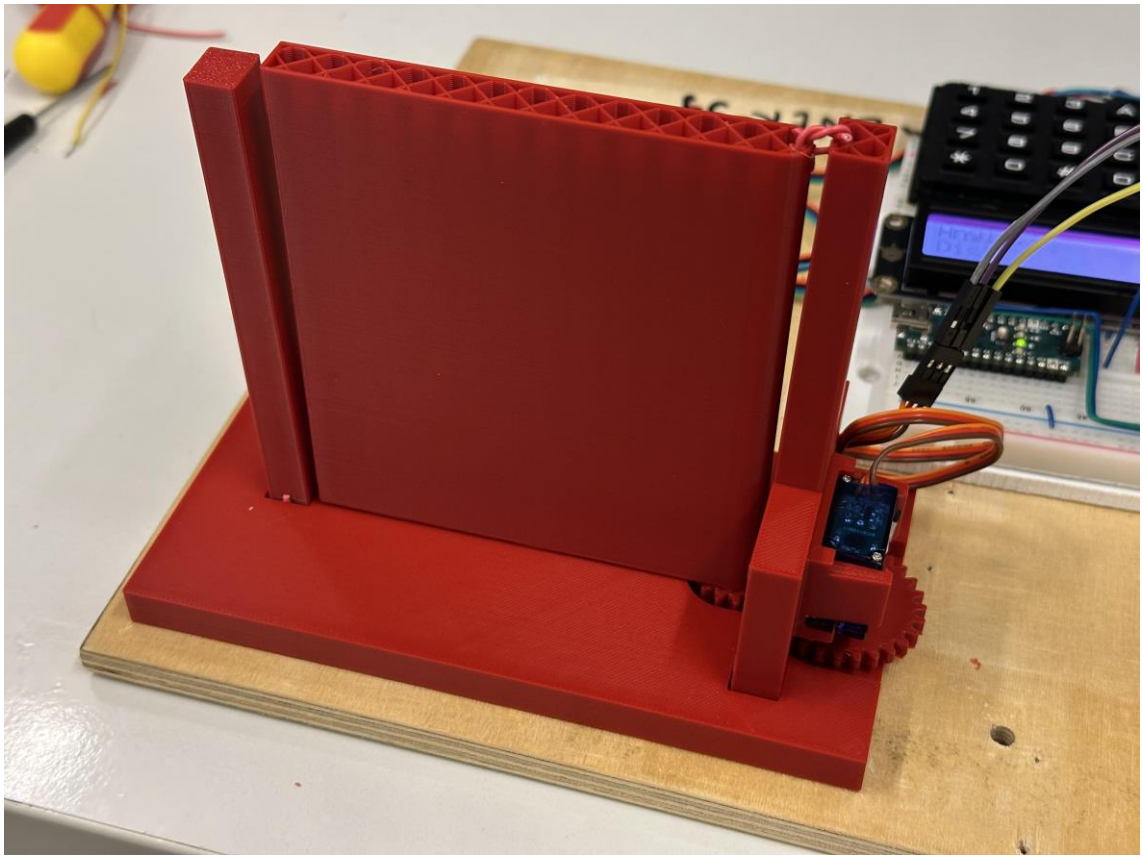
KUVA 2. Kokonaisuus



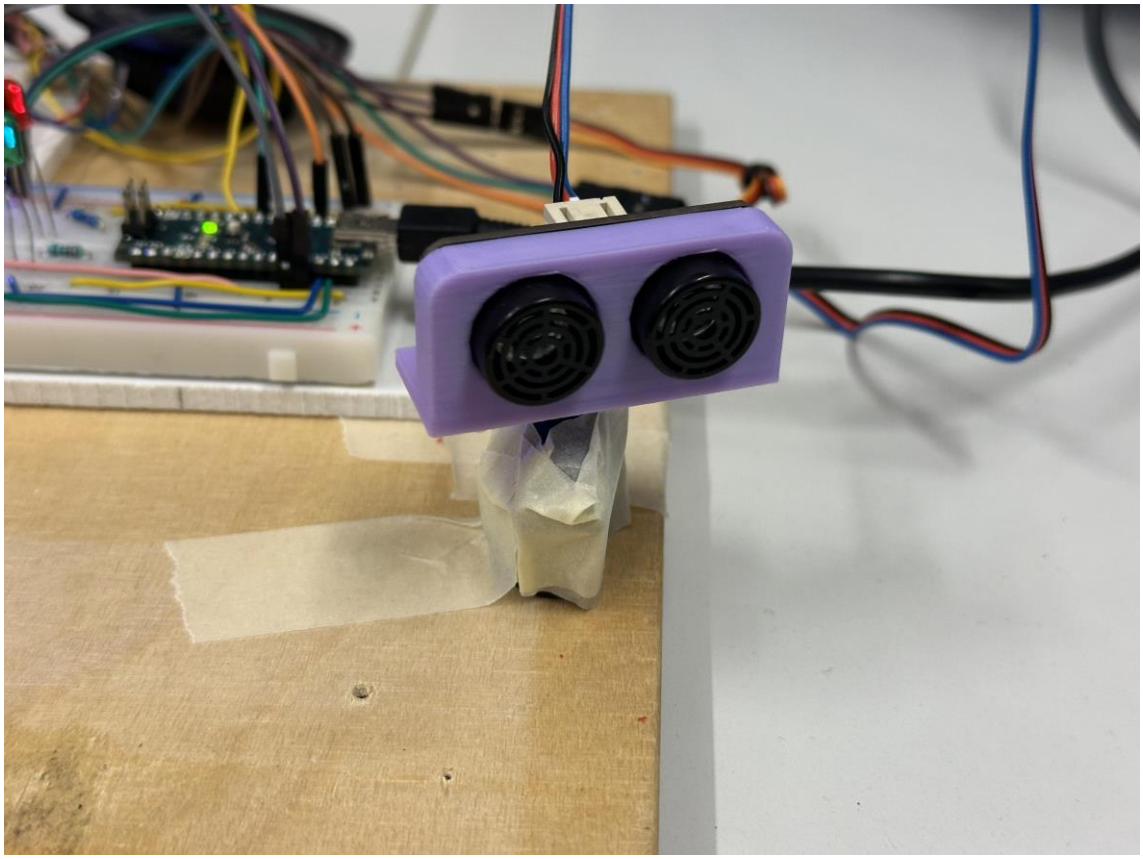
KUVA 3. Pääalusta, Nanot ja muut pienkomponentit



KUVA . Näppäimistö ja näyttö



KUVA 5. Ovimekanismi



KUVA 6. Tutka

4.5 Testaus/toiminta

Järjestelmän testaaminen oli aika yksiselitteistä. Näyttöön päivittyy jatkuvasti tutkan havaitsema etäisyys, sekä tutkan tämänhetkinen kulma. Kun kättä vilauttaa n.10 cm päästä tutkasta, tutkan servo pysähtyy, tulee äänimerkki, ja näyttöön tulee punaisella taustalla teksti “Enter Code:”, jossa se kysyy pääsykoodia ovesta sisään. Tästä eteenpäin voi tapahtua muutama erilainen skenaario:

1. Jos koodin antaa oikein, näyttöön tulee teksti “Access Granted” vihreällä taustalla, soi kaiuttimesta hyväksymismelodia, ja tämän jälkeen ovi aukeaa, näytössä lukien “Door Ready”. Kuluu n.10 sekuntia, jonka jälkeen ovi menee kiinni, näytössä lukee “Door Closed”, ja järjestelmä kysyy pääsykoodin vaihtoa, haluaako sen vaihtaa johonkin muuhun tekstillä “Change Pass?” “Type # Or **”. “#” merkillä salasanan voi uusia, kun taas “**” salasana pysyy ennallaan. Tämän jälkeen tutka alkaa taas liikkua, ja jatkaa vahtimistaan.

2. Jos koodin antaa väärin, tulee oma äänimerkkinsä sille, ja näyttöön teksti “Wrong Code” “Try Again!”, ja sen jälkeen pyytää koodia uudestaan. Kolmen väärän yrityksen jälkeen järjestelmä lukitustilaan minuutiksi, jolloin näyttöön tulee ensin ilmoitus “Too Many Wrong Attempts” “Doorpad Is Closed For 1 Minute” ja tämän jälkeen näyttöön ilmestyy laskuri, joka laskee jäljellä olevat sekunnit. Kun tämä on mennyt ohi, järjestelmä siirtyy tutkailutilaan, jolloin koodin pyyntö on aktivoitava uudestaan.

3. Jos odottaa tekemättä mitään 7 sekuntia, näyttöön tulee teksti “Timeout!” ja järjestelmä palaa jatkamaan ympäristön tutkailua. Tämä toiminto on myös toiminnassa muissa koodin syötön vaiheissa.

Järjestelmä toimii näiden suoritusten aikana ilman mitään ongelmia. Jos jostain syystä järjestelmä menisi jumiin, vahtikoira keskeyttäisi toiminnan ja aloittaisi sen alusta, jolloin oven asentokin on vakiona kiinni. Ohessa vielä linkki videoon, josta näkee tutkan toiminnan: 20251209_131230000_iOS.MP4

5 POHDINTA

Harjoitustyön teko alusta asti omatoimisesti ilman valmista pohjaa kurssilta toi uuden kulman sulautetun järjestelmän rakentamiseen. Yhtenä ryhmän isoimpana haasteena oli keksiä itse aihe mitä rakentaa. Lopulliseen työhön otimme inspiraatiota netistä löytyvistä tutkamalleista, mutta halusimme silti rakentaa oman alustan, sekä lisätä siihen vielä muita ominaisuuksia. Isoimpana oppimiskohteena oli eri osien integroinnin selvittely. Monet komponenteista oli kylläkin tuttuja, juuri samalla kurssilla käytettyjä, mutta niiden käyttöönotto eri toimintojen yhteydessä laittoi hieman ryhmäläisiä miettimään miten niiden kanssa onnistuisi.

Yhden henkilön puuttuminen osan ajasta työtä vielä vähän hankaloitti työntekoa, jonka takia joitain viimeistelyitä ei saatu ihan valmiiksi. Myös 3D-tulostuksen kanssa syntyi joitain ongelmia, joka näkyy esimerkiksi oven viimeistelyssä negatiivisesti, sekä puutteellisissa kiinnityksissä. Saimme siitä huolimatta toteutettua kokonaisen ja toimivan sulautettujen järjestelmän piiriratkaisun, jota olisi mahdollista jatkojalostaa isompaankin skaalaan. Poissaolo poisluettuna kaikki tiimin jäsenet olivat mukana tekemisessä tasapuolisesti, kommunikointi oli sujuvaa, ja kaikilta löytyi hyvä motivaatio hieman haastaa itseään tämän projektin puitteissa. Olemme itse tyytyväisiä lopputulokseen, sekä uuden oppimiseen, jota projektin aikana tapahtui.