Serra Idroponica Smart

Giovanni Pantarotto 157707
Enrico Contessi
Cristian Tomass

6 ottobre 2025

Corso di Internet Of Things IBML

1 Introduction

La serra idroponica è una serra diversa dalle altre, sfrutta un flusso d'acqua costante arricchito di nutrienti dal quale le radice delle piante, che vi entrano a contatto, ricavano i sali necessari per crescere.

2 Hardware

2.1 Componenti utilizzate

- Raspberry Pi 4
- Arduino Uno
- Pompa sommersa
- 3 Pompe peristaltiche
- Sensore Temperatura
- Sensore EC
- Sensore PH
- Sensore livello dell'acqua
- Strisce LED blue/red
- Relè
- Elettrovalvola
- Pulsante d'emergenza
- Trasformatore 220V to 3.3V/5V/12V

3 Software

Codice Arduino Codice per la gestione degli stati

```
#include "stati.h"
#include <Arduino.h>

#define valve 5      //electrovalve pin

// #include <NewPing.h>   //sonar library
// #define pin_trig 9            //sonar pins
```

```
// #define pin_echo 6
// #define massimo 1000        //max distance
// NewPing sonar(pin_trig, pin_echo,  massimo);   //initialize sonar

// Water level sensor
#define PIN_SIGNAL_WL A5
#define POWER_PIN 6
#define massimo 1000


#include <OneWire.h>                //libraries for DS18B20 temperature probe
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 7    //temp probe pin
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);     //initialize T probe

// PH sensor
#define PH_PIN = A1    //pH pin


#define EC_PIN A2      // ec pin
#define VREF 5.0      //EC analog reference voltage(Volt) of the ADC
#define SCOUNT  30           //EC sum of sample point
int analogBuffer;    // store the EC analog value in the array, read from ADC
float averageVoltage = 0, tdsValue = 0;



#define tankH 30   //tank height in cm
#define tankL1 30    //l1 in cm
#define tankL2 30   //l2 in cm
#define totVol ((tankH-5)*tankL1*tankL2)/1000 //tank volume in L
#define concentration 100  //nutrient SOL A and B  concentration
#define correctionSpeed 70   //correction speed (%)
#define perpumpA 10  //peristaltic pump pins
#define perpumpB 12
#define pHpumpC 11
#define dosingUnit11 1.4996    // measured reach in ml/sec of 3 peristaltic pumps
#define dosingUnit13 1.5185
#define dosingUnit12 1.4826
#define pHoffset 0.2
#define ECoffset 0.05
int solAvol = (totVol*1000) / concentration;    //mL     for solution adjustment
int solBvol = (totVol*1000) / concentration;    //mL
int h2oVol = (totVol*1000) - (solAvol + solBvol);   //mL



void go(struct app_state *st, enum stati dest){    //function to change state, defined
     in stati.h
  st->current = dest;
  st->first = true;
}

void app_level(struct app_state *st){   //function to get tank level
  if (st->first) {
    Serial.println("get level");          //in every state write once the state
    function's name, controlled by bool first defined in struct in stati.h
    st->first = false;
  }

  digitalWrite(valve, st->valveclosed);    //make sure valve is closed

  int  lettura = analogRead(PIN_SIGNAL_WL);
  st->level = ((tankH - lettura) * tankL1 * tankL2) / 1000;   //read distance between
    surface and sonar and convert to L
  //Serial.print(lettura);
  //Serial.println(" cm");
  Serial.print(st->level);
  Serial.println(" L");
```

```
 if(st->level < totVol ){   //check if level must be restored
  go(st, st_fill);         //if yes go to fill state
  } else
  go(st, st_temp);
}



void app_fill(struct app_state *st){
  if (st->first) {
    Serial.println("fill the tank");
    st->first = false;
  }

  while (st->level < totVol) {                        //open valve until level is
    restored
    digitalWrite(valve, !st->valveclosed);
    int lettura = analogRead(PIN_SIGNAL_WL);
    st->level = ((tankH - lettura) * tankL1 * tankL2) / 1000;
   }
    go(st, st_level);

}



void app_temp(struct app_state *st){          //get temperature function
  if (st->first) {
    Serial.println("get temperature");
    st->first = false;
  }
  sensors.requestTemperatures();
  st->  T = sensors.getTempCByIndex(0);

  Serial.print("temperature: ");
  Serial.println(st->T);
  go(st, st_ph);
}




void app_ph(struct app_state *st){          //get ph function
  if (st->first) {
    Serial.println("get pH");
    st->first = false;
  }
   st->ph=pH.read_ph();
  Serial.print("pH: ");
  Serial.println(pH.read_ph());
  go(st, st_ec);
}

void app_ec(struct app_state *st){        //get EC function
  if (st->first) {
    Serial.println("get  ec");
    st->first = false;
  }

  analogBuffer = analogRead(EC_PIN);    //read the analog value and store into the
    buffer
  averageVoltage = analogBuffer * (float)VREF / 1024.0; // read the analog value more
    stable by the median filtering algorithm, and convert to voltage value
  float compensationCoefficient = 1.0 + 0.02 * (st->T - 25.0); //temperature
    compensation formula: fFinalResult(25^C) = fFinalResult(current)/(1.0+0.02*(fTP
    -25.0));
  float compensationVolatge = averageVoltage / compensationCoefficient; //temperature
    compensation
  tdsValue = (133.42 * compensationVolatge * compensationVolatge * compensationVolatge
     - 255.86 * compensationVolatge * compensationVolatge + 857.39 *
    compensationVolatge) * 0.5; //convert voltage value to tds value
```

```
  //Serial.print("voltage:");
  //Serial.print(averageVoltage,2);
  //Serial.print("V   ");
  //Serial.print("TDS Value:");
  //Serial.print(tdsValue,0);
  //Serial.println("ppm");
  st->ecValue = tdsValue / 640;    //convert tds to ec
  Serial.print("EC: ");
  Serial.println(st->ecValue);
  go(st, st_adj);
}



void app_adj(struct app_state *st){              //adjust solution
  if (st->first) {
    Serial.println("adjust nutrient solution");
    st->first = false;
  }
  float deltaPH = st->ph - (st->pHsetpoint - pHoffset);           //calculate pH delta
  float deltaEC = (st->ECsetpoint - ECoffset) - st->ecValue;          //calculate EC
    delta
  Serial.print("delta EC: ");
  Serial.println(deltaEC, 2);
  Serial.print("delta pH: ");
  Serial.println(deltaPH);
  //   PROPORTION:  VTot=10L ; concentration 50:1   --->   solAvol / ECsetpoint = x /
    deltaEC;
if (deltaEC > ECoffset || deltaPH > pHoffset ) {                  //check if any
    correction is needed
    if (deltaEC > ECoffset){
    float  solA_qty = (deltaEC * solAvol) / (st->ECsetpoint);         //calculates
    SolA+B qty needed to adjust
    float  solB_qty = (deltaEC * solBvol) / (st->ECsetpoint);
    float dosingA = solA_qty * (1 / dosingUnit13);            //convert quantity in
    pumping time
    float dosingB = solB_qty * (1 / dosingUnit12);
    dosingA = (dosingA / 100) * correctionSpeed;        //adjust time for correction
    speed
    dosingB = (dosingB / 100) * correctionSpeed;
   /*
    Serial.print("EC set point:");            //debugging
    Serial.println( ECsetpoint - ECoffset);
    Serial.print("qta' sol A: ");
    Serial.println(solA_qty, 2);
    Serial.print("qta' sol B: ");
    Serial.println(solB_qty, 2);
    Serial.print("tempo dosaggio sol A: ");
    Serial.println(dosingA, 0);
    Serial.print("tempo dosaggio sol B: ");
    Serial.println(dosingB, 0);
    digitalWrite(perpumpA, HIGH);               //activate pumps for the time needed
    Serial.println("SolA dosing");
    delay(dosingA * 1000);
    delay(1000);
    digitalWrite(perpumpA, LOW);
    Serial.println("SolA stop");
    delay(300);
    digitalWrite(perpumpB, HIGH);
    Serial.println("SolB dosing");
    delay(dosingB * 1000);
    delay(1000);
    digitalWrite(perpumpB, LOW);
    Serial.println("SolB stop");
    delay(300);
  }  if ( deltaPH > pHoffset ) {
    digitalWrite(pHpumpC, HIGH);
    Serial.println("Acid dosing");          //activate acid pump for 1 sec
    delay(1000);                          //change this value empirically based on type
    of acid, total volume, correction time
    digitalWrite(pHpumpC, LOW);
```

```
    Serial.println("Acid stop");
  }
  go(st, st_ph);
 } else go(st, st_level);
}
```

Libreria per la gestione degli stati

```
#ifndef _STATI_H_
#define _STATI_H_

#include <Arduino.h>

enum stati {st_level, st_fill, st_temp, st_ph, st_ec, st_adj};     //name different
    states

struct app_state {                          //data packet to pass and modify in the
    whole code
  enum stati current;
  bool first = true;
   float level;
   float T;
   float ph;
   float ecValue;
   float ECsetpoint=1.5;     //  desired EC and pH
   float pHsetpoint= 5.8;
   bool valveclosed=0;

};

void go(struct app_state *st, enum stati dest);         //go to another state function

void app_level(struct app_state *st);        //functions defined, code is in stati.cpp
void app_fill(struct app_state *st);
void app_temp(struct app_state *st);
void app_ph(struct app_state *st);
void app_ec(struct app_state *st);
void app_adj(struct app_state *st);


#endif
```

4 Testing

Comments can be added to your project by highlighting some text and clicking “Add comment” in the top right of the editor pane. To view existing comments, click on the Review menu in the toolbar above. To reply to a comment, click on the Reply button in the lower right corner of the comment. You can close the Review pane by clicking its name on the toolbar when you’re done reviewing for the time being.

Track changes are available on all our premium plans, and can be toggled on or off using the option at the top of the Review pane. Track changes allow you to keep track of every change made to the document, along with the person making the change.

4.1 How to add Lists

You can make lists with automatic numbering . . .

1. Like this,
2. and like this.

. . . or bullet points . . .

- Like this,
- and like this.

4.2 How to write Mathematics

LaTeX is great at typesetting mathematics. Let $X_1, X_2, \ldots, X_n$ be a sequence of independent and identically distributed random variables with $\mathrm{E}[X_i] = \mu$ and $\mathrm{Var}[X_i] = \sigma^2 < \infty$, and let

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n}\sum_{i}^{n} X_i$$

denote their mean. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$.

4.3 How to change the margins and paper size

Usually the template you're using will have the page margins and paper size set correctly for that use-case. For example, if you're using a journal article template provided by the journal publisher, that template will be formatted according to their requirements. In these cases, it's best not to alter the margins directly.

If however you're using a more general template, such as this one, and would like to alter the margins, a common way to do so is via the geometry package. You can find the geometry package loaded in the preamble at the top of this example file, and if you'd like to learn more about how to adjust the settings, please visit this help article on page size and margins.

4.4 How to change the document language and spell check settings

Overleaf supports many different languages, including multiple different languages within one document.

To configure the document language, simply edit the option provided to the babel package in the preamble at the top of this example project. To learn more about the different options, please visit this help article on international language support.

To change the spell check language, simply open the Overleaf menu at the top left of the editor window, scroll down to the spell check setting, and adjust accordingly.

4.5 How to add Citations and a References List

You can simply upload a `.bib` file containing your BibTeX entries, created with a tool such as JabRef. You can then cite entries from it, like this: [**?**]. Just remember to specify a bibliography style, as well as the filename of the `.bib`. You can find a video tutorial here to learn more about BibTeX.

If you have an upgraded account, you can also import your Mendeley or Zotero library directly as a `.bib` file, via the upload menu in the file-tree.

4.6 Good luck!

We hope you find Overleaf useful, and do take a look at our help library for more tutorials and user guides! Please also let us know if you have any feedback using the Contact Us link at the bottom of the Overleaf menu — or use the contact form at `https://www.overleaf.com/contact`.