Open Source Automation for Hydroponics

Design, Construction, Programming and Testing

Anton Sundgren

Bachelor's Thesis
February 2015
Environmental Engineering

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Degree Programme in Environmental Engineering

ANTON SUNDGREN:
Open Source Automation for Hydroponics
Design, Construction, Programming and Testing

Opinnäytetyö 34 sivua

Helmikuu 2015

Tämän opinnäytetyön tarkoituksena on mahdollistaa automatisoidun vesiviljely-järjestelmän rakentaminen käyttäen avoimen lähdekoodin laitteistoa ja ohjelmistoa (Arduino). Tämän työn tuloksia voidaan käyttää ohjeena tämäntyyppisen järjestelmän rakennusprosessissa.

Laitteiston päätehtävänä on automatisoida ravinneliuoksen pH ja sähkönjohtavuus-arvojen säädön seuramaalla ja tarvittaessa manipuloimalla niitä. Tämä saavutetaan käyttäen useita avoimen lähdekoodin ja laitteistoin ohjainmoduuleja, kuten esimerkiksi Arduino Mega2560, MiniPH, EC ja DC-moottori ohjain.

Tässä työssä käydään läpi jokaisen moduulin perustoiminnot ja esitetään pieni pätkä lähdekoodia jolla ohjataan kyseisen moduulin toimintoja. Myös testauskokoonpano on esitetty ja tiedonkeruusta saadut tulokset.

Työssä esitetään myös matemaattiset kaavat, joiden avulla pH:n ja sähkönjohtavuuden lämpötilakorjaukset tehdään.

Kehitetty järjestelmä testattiin TAMK:in kasvihuoneessa. Järjestelmä oli käynnissä 21 päivää, jonka aikana mittaustulokset tallennettiin SD kortille. Kerätyn aineiston tarkempi analyysi osoittaa, että laite toimii odotetusti.

Asiasanat: Arduino mega 2560, automaatio, vesiviljely, pH, sähkönjohtavuus, avoin lähdekoodi

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Environmental Engineering

ANTON SUNDGREN:
Open Source Automation for Hydroponics
Design, Construction, Programming and Testing

Bachelor's thesis 34 pages

February 2015

The purpose of this thesis is to provide a framework for constructing automated hydroponic system using open source hardware and software (Arduino). This work can be used as a guide for constructing such system.

The main function of the hardware is to automate the measurement and adjustment of the pH and electric conductivity of the nutrient solution used to feed the tomato plants. This is accomplished using multiple open hardware modules, such as Arduino Mega2560 board, MiniPH interface, EC interface and DC-motor interface.

This thesis describes the basic function of each module and provides some source code to run those modules; it also describes the test setup, data collection and obtained results. The mathematical functions used for calibration and temperature compensation of pH and EC (Electric conductivity) are also presented. The developed system was tested in TAMK’s greenhouse. The system was running for 21 days during which parameter data was recorded. The collected data and the analysis of the results concluded that the system was operating as expected.

Key words: Arduino mega 2560, automation, hydroponics, pH, conductivity, open source

SISÄLLYS

1 INTRODUCTION

In hydroponic systems the plant productivity is closely related with to nutrient uptake and the pH regulation (Marchner, 1995). The pH level of the nutrient solution affects the availability of the nutrients to plants (T.Asao, 2012). Unmonitored pH level of the nutrient solution will quickly lead to a situation where plants can't absorb the essential nutrients, if not corrected this will eventually lead to unhealthy plant growth and poor productivity. Regulation of pH is normally carried out by using nitric, sulphuric or phosphoric acid, in this experiment the diluted phosphoric acid (~20%) was used.

A nutrient solution for hydroponic system is an aqueous solution containing mainly inorganic ions from soluble salts of essential elements for higher plants. The total ionic concentration of a nutrient solution determines the growth, development and production of plants (T.Asao, 2012). Electrical conductivity of the nutrient solution can be used as a good indicator of the amounts of available ions to the plants in the root zone (T.Asao, 2012). The control of nutrient solution concentration is achieved by adding more water to the solution. The possibility of controlling the concentration of the nutrient solution enables feeding the plants with the most optimal amount of nutrients. The optimal concentration varies with the plant species and plants developmental stage.

The constant monitoring and manipulation of pH and EC is essential for good plant growth. This can be easily archived using computerized commercial system but because of the high price and big scale design it is impractical for a small scale grow setup. Designing this kind of system is considered to be impractical and also highly expensive requiring the excellent knowledge of electronic and programming, but thanks to open source hardware and software the knowhow is now easily available to everyone.

This work approaches the problem of building such system by implementing modular design which is built around Arduino open source hardware and software. Utilizing pre manufactured modules lowers the price and the learning curve; it also provides easily accessible source code which can be modified to suit the needs of this project.

2 HARDWARE

2.1 ARDUINO

Arduino is and open source hardware and software prototyping platform based on simple and easy to use software and hardware. Currently there are millions of projects which use Arduino as the main component, it is used by artist, hackers, hobbyist, teachers and also professionals (Borchers,2013).

2.1.1 Arduino Mega 2650r3

Arduino Mega2560 (Figure 1) is a microcontroller board which it is based on the ATmega2560 microprocessor. This MCU has 256KBytes of In-System Self-Programmable Flash memory, 4Kbytes EEPROM, 8Kbytes Internal SRAM, 54 digital input/ output pins of which 15 can be used as PWM (Pulse-width modulation) outputs, USB connector, 16 analog inputs and it runs at 16MHz (Atmel, 2014). The board operates on an external supply of 6 to 20v and it can be used as a DC/DC convertor to produce 5V and 3.3V output for external devices (Arduino.cc, 2013).

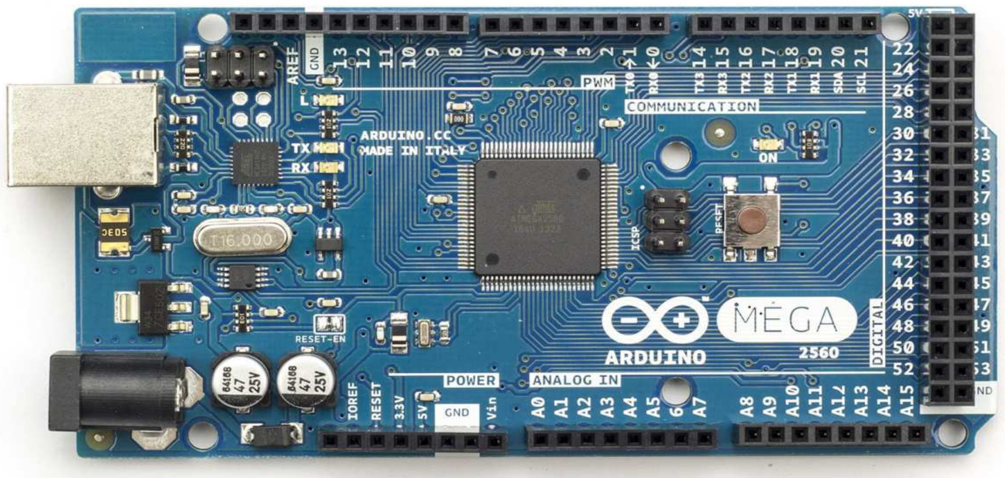


Figure 1. Arduino Mega 2560 board (Arduino.cc , 2014)

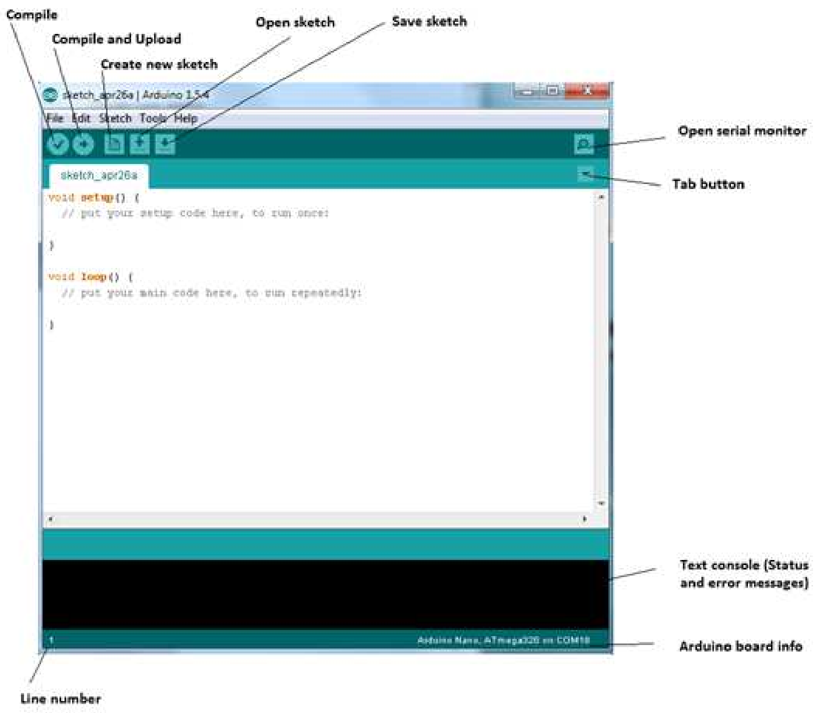
Some of the pins on the board serve additional functions (Arduino.cc, 2013):

- Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data.
- PWM: 2 to 13 and 44 to 46. Provide 8-bit PWM output.
- SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS)
- LED: 13. There is a built-in LED connected to digital pin 13.
- TWI: 20 (SDA) and 21 (SCL). Two wire Interface

The USB connector can be use to connect the device to computer for programming or monitoring purposes.

2.1.2 Arduino Integrated Development Environment (IDE)

Arduino IDE (Figure 2) is an open source environment which is design to simplify the C and C++ code writing, debugging and flashing. It runs on MAC OS, Linux and Windows and is written in Java and based on Processing, avr-gcc, and other open source software (Arduino.cc, 2013)



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure 2. Arduino IDE environment

Arduino IDE is used to write, debug and upload the code into the microprocessor located on the Arduino board. The main function of Arduino IDE is to turn written C/C++ code into machine code that Arduino can execute.

3 SYSTEM DESIGN COMPONENTS

The aim of the project was to construct an automated hydroponics controller which would measure air temperature, water temperature, humidity, pH and EC. To control the pH and EC the controller would control 2 peristaltic pumps, one for adding acid (to decrease the pH) and one to add more water (to decrease the conductivity). To achieve all of this functionality prebuilt modules were used, these were: pH, EC and motor driver.

3.1 pH interface

The pH measurement interface (Figure 3) was ordered from “Sparky’s widgets” website. The pH module was chosen based on the low cost, good functionality and low physical size. The module measured the output voltage of the pH electrode, amplified it and converted the analog value to digital one using 12bit analog to digital converter.

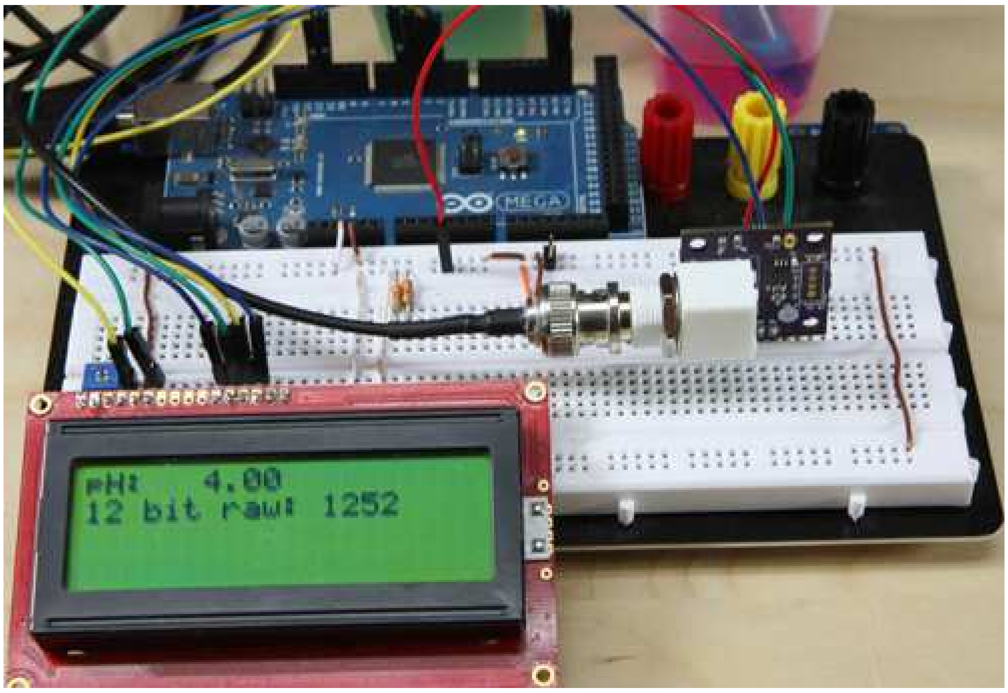


Figure 3. The MinipH pH (Small board with BNC connector) interface in action (Sparkyswidgets, 2014)

3.2 EC interface

The EC measurement interface (Figure 4) was ordered from “Practical Maker” website. This circuit was chosen because of low price, simple design and low cost. It had also good measuring precision and could be used with wide variety of EC electrodes (K0.1-K10) by small modification in the electronics and code.

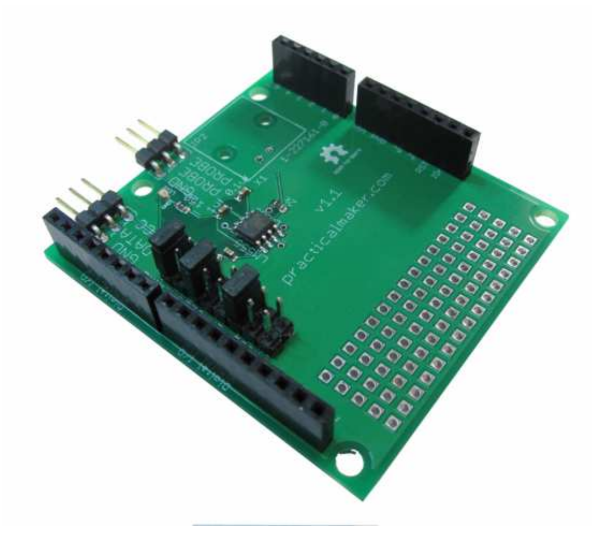


Figure 4. ”Practical maker” EC Shield (Practicalmaker, 2014)

3.3 Temperature and humidity sensors

The temperature was measured by two sensors DS18B20 (Maxim Integrated) and STH11 (Sensirion), one of the sensors (DS18B20) was waterproof and was used to measure the temperature of liquid (nutrient solution) and the other (SHT11) was used to measure temperature of the surrounding air and also the humidity of the air (Figure 5).
The DS18B20 was chosen because of the low price and good accuracy (±0. 5°C). The SHT11 was chosen because it had two onboard functionalities; it could measure temperature and humidity.

Figure 5. Waterproof DS18B20 sensor on the left and SHT11 on the right (Adafruit, 2014 . Sensirion, 2011)

3.4 DC motor interface

The “Adafruit Motor Shield” (Figure 6) was chosen based on the ability to run four 6v DC motors, small physical size and low cost. The shield was ordered from Adafruit.com. This module was used to control the peristaltic pumps.

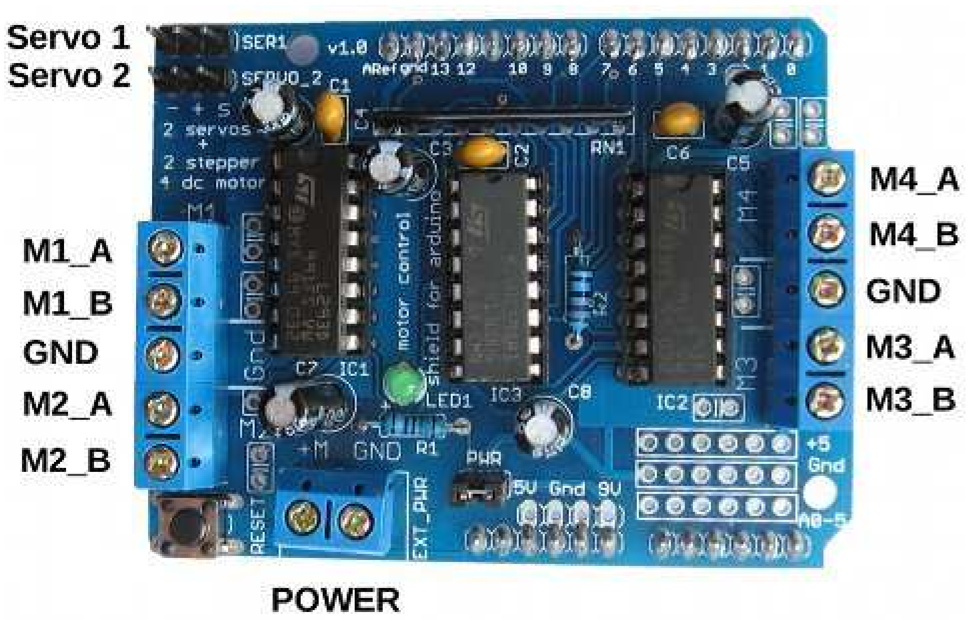


Figure 6. Adafruit motor shield (Adafruit, 2014)

3.5 SD module

Simple SD module (Figure 7) was used to enable data logging on the device. The module uses micro SD card and communicates with Arduino using SPI interface.

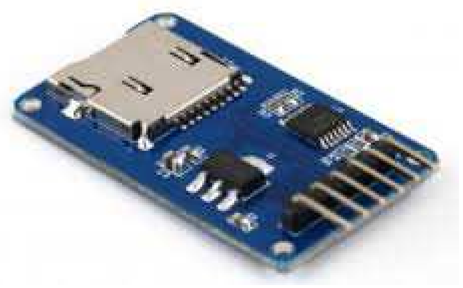


Figure 7. Adafruit motor shield (Adafruit, 2014)

3.6 Other hardware

Other hardware used in the project:

- 1.8" TFT LCD module with serial interface (Figure 6 left). 128x160 Resolution and 262K colors (Figure 8 left).
- Small peristaltic pump (4pcs)(Figure 6 right). 20-60ml/min flow rate.
- On/ Off button
- Power connector
- Plastic enclosure to house all of the parts
- Wires to connect the devices
- pH electrode
- EC electrode (K=0.1)
- Tubing

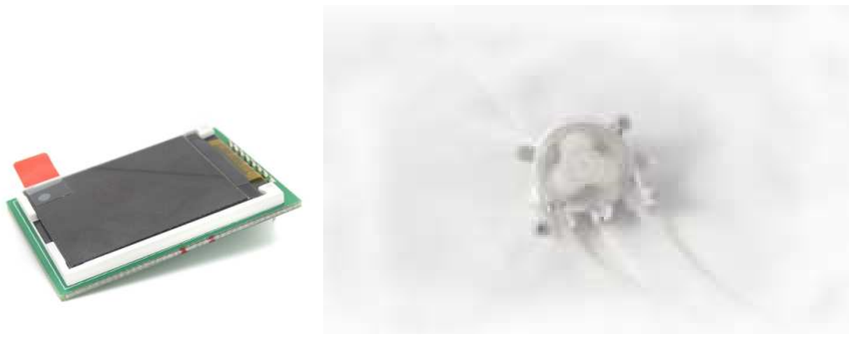


Figure 8. TFT screen (Leftt) and the miniature peristaltic pump (Right) (Sundgren, 2014)

4 THE CONSTRUCTED SYSTEM

Arduino MEGA 2560 is used as a sensor node, it requests, and process sensor data. Most of the parts were connected to the Arduino using simple jumper wires but some of the wires were soldered to ensure that they won't get loose. SPI, One wire, PWM and digital I/O ports were used to connect modules to the main unit (Figure 9).All of the electronics were then placed into plastic enclosure (Figure 10) to protect delicate electronics from dust and moisture.

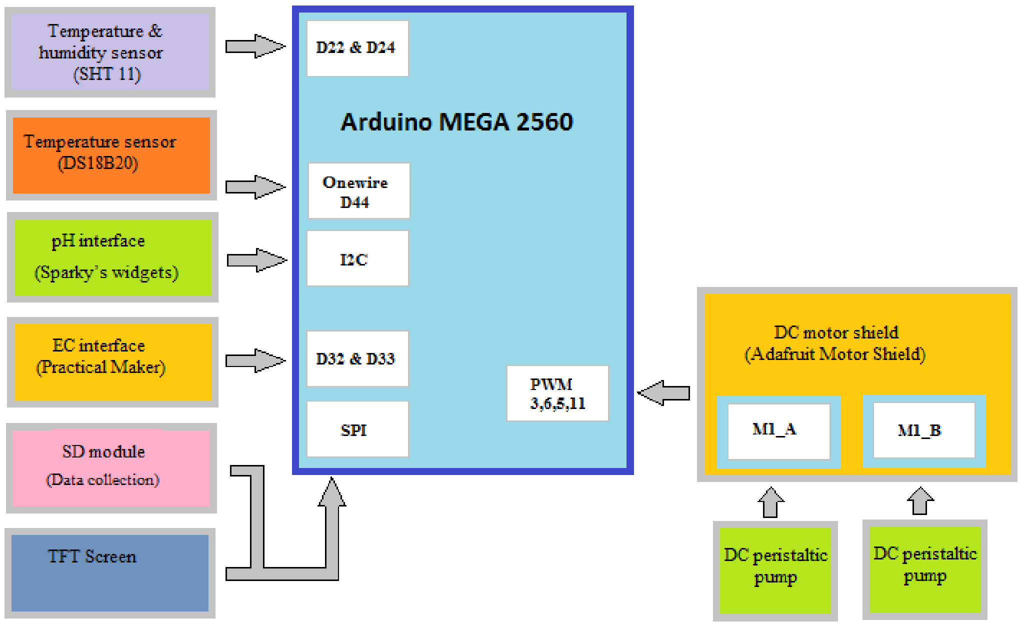

Temperature & humidity sensor (SHT 11)
Temperature sensor (DS18B20)
pH interface (Sparky's widgets)
EC interface (Practical Maker)
SD module (Data collection)
TFT Screen
D22 & D24
Arduino MEGA 2560
Onewire D44
I2C
D32 & D33
SPI
PWM 3,6,5,11
DC motor shield (Adafruit Motor Shield)
M1_A
M1_B
DC peristaltic pump
DC peristaltic pump

Figure 9. Overview of the connection ports used by the prototype made in this project

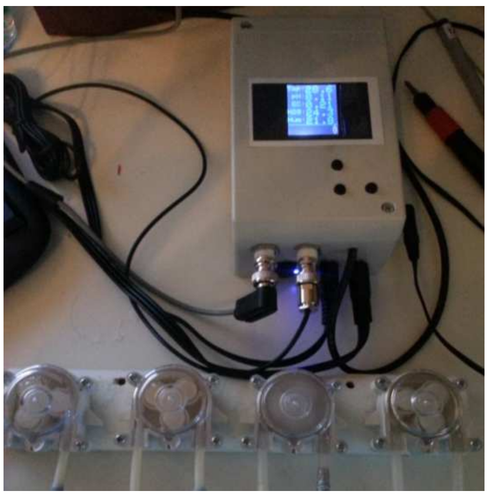


Figure 10. Plastic enclosure used to protect the electronics

The peristaltic pumps were attach to the wooden frame and connected to the device using wires. Only two of the pumps were used. The prototype was powered using 9v DC power supply.

5 PROGRAMMING

Some of the code was obtained from the open source projects and modified to fulfill the needs of this project, currently the code is 1833 lines long.

5.1 Libraries

The Arduino environment can be extended through the use of libraries, just like most programming platforms. Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data (Arduino.cc, 2013).

Following libraries were used in the code:

- Adafruit_GFX.h (TFT screen)
- Adafruit_ST7735.h (TFT screen)
- Eeprom.h (Saving calibration data to Electronically Erasable Programmable Read-Only Memory)
- Math.h (Mathematical functions for manipulating floating point numbers)
- SHT1x.h (Provides a simple interface to the SHT1x series temperature / humidity sensors from Sensirion)
- SD.h (Reading from and writing to SD cards)
- Servo.h (Controlling motors)
- SPI.h (Communicate with SPI devices. SD shield and TFT screen)
- String.h (Manipulation of text strings)
- OneWire.h (One wire library for DS18B20 temperature sensor)
- Statistic.h (Calculation of standard deviation and running average for pH and EC)
- Wire.h (I2C communication library for pH circuit)

5.2 Calibration

Essentially both pH and EC calibration code works in a similar way. The slope of the line is determined from slope equation using two calibration points; $pH_1$is simply 4 (pH4 calibration) and pH_2 is 7 (pH7 calibration). E_1 is the electrical potential at pH4 and E_2 is the potential measured at pH7.

$$pH_{slope} = \frac{E_2 - E_1}{pH_2 - pH_1}$$

The calibration for EC is done in similar way; $EC_1$is 1413µS, EC_2 is 220µS and f_1 is measured frequency (Hz) at EC_1 and $f_2$is the measured frequency (Hz) at EC_2.

$$EC_{slope} = \frac{f_2 - f_1}{EC_2 - EC_1}$$

5.3 Measuring pH and temperature correction

The measurement of pH is started by converting the analog signal received from the "Mini pH" circuit to 12 bit digital value using analog to digital converter (ADC), this value is then converted to the actual pH value using two equations. Equation 1, converts the 12bit digital value to mill volts (mV) and second equation calculates the pH value based on the slope of the calibration line and mV.

$pH_{7cref} = pH7\ calibration\ point\ 12bit\ value(0 - 4096)$

$pH_{raw} = 12bit\ value(0 - 4096)$

$V_{ref} = 4{,}096v\ (Voltage\ reference)$

$Gain = 5{,}25\ (Gain\ of\ the\ amplification\ circuit)$

$pH_{init} = Initial\ pH$

$pH_{corrected} = pH\ value\ after\ temperature\ correction$

$temp = measured\ air\ temperature\ in\ °C$

$pH_{slope} = Slope\ of\ the\ pH\ calibration\ line$

Equation 1. (Sparkyswidgets,2013)

$$mV = \frac{\frac{V_{ref} * pH_{7cref}}{4096} * 1000 - \frac{pH_{raw}}{4096} * V_{ref} * 1000}{Gain}$$

Equation 2. (Sparkyswidgets,2013)

$$pH_{init} = 7 - (\frac{mV}{pH_{slope}})$$

As the actual pH is temperature dependent it is necessary to apply the temperature compensation using equation 3 (Pasco, 2001).

Equation 3. (Pasco, 2001).

$$pH_{corrected} = \frac{(pH_{init} - 7) * (273.15 + temp)}{(273.15 + 25)} + 7$$

After the temperature compensation the displayed pH value corresponds to the pH value measured at 25°C no matter at what temperature the measurement was made in thus eliminating the error caused by the temperature difference.

5.4 Measuring EC and temperature correction

The conductivity measurement is started by calculating the frequency received through the predefined pin (in this case pin 30). This is fully digital pin thus it is measuring the change of state from “1” to “0” and back, “1” being roughly +5v and “0” 0v.

As the conductivity in the sample rises the time between the changes of the state from “1” to “0” shortens thus increasing the frequency. This difference in frequency is measured using “pulseln” command.

As the pin goes high the timer is started and when the pin goes low the timer is stopped, this is repeated 125 times to improve the precision of the reading. Each time the calculated delay is added to the variable which represents the total measured time during 125 measurements.
There is also second “pulseln” command which is also monitoring the state but only in opposite way; the timer is started when the state of the pin goes low and stops when it goes back high. In this way length of the both states is measured and thus the full wavelength is captured (Figure 11).

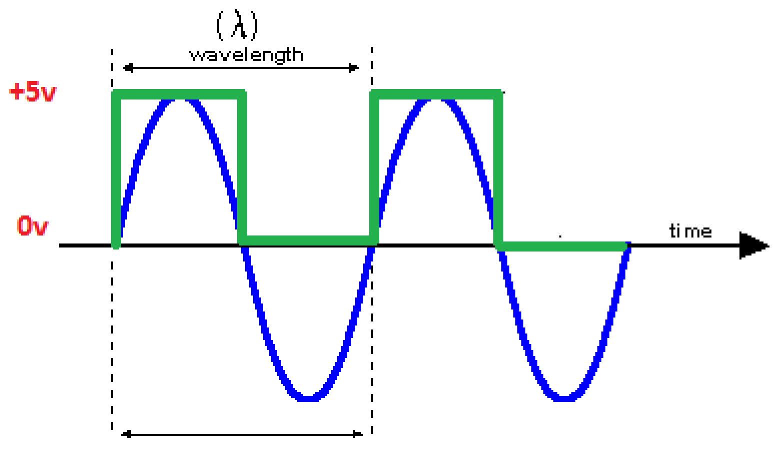


Figure 11. The green line representing the one full measurement cycle from low to high and back

Next the frequency (f_{init}) is calculated using following equation:

Equation 4 (Practicalmaker,2014).

$$samples = 125$$

$$f_{init} = \frac{100000}{\left(\frac{f_h}{samples}\right) + \left(\frac{f_l}{samples}\right)}$$

The slope of the calibration line is calculated using slope equation;

$$EC_{slope} = \frac{f_2 - f_1}{EC_2 - EC_1}$$

The actual conductivity value is calculated using linear regression equation:

$$y = a + bX$$

Where:

$y = value\ in\ \mu S\ (EC_{Ti})$

$x = measured\ frequency$

$b = slope\ of\ the\ line(EC_{slope})$

$a = y\ interception$

$S_x = (X_1 + X_{2)}$

$S_y = (Y_1 + Y_2)$

$a = S_y - (b * S_x)$

As the conductivity is highly temperature dependent (M.Mäntynen,2001) it is necessary to apply the temperature compensation using equation 4.

Equation 4. (M.Mäntynen,2001)

$$EC_{25^\circ C} = \frac{EC_{Ti}}{1 + \left(\frac{\theta}{100}\right) * (T_i - T_{25^\circ C})}$$

$\theta = Temperature\ compensation\ factor = 1.885\%/°C$

$T_i = measuring\ temperature$

$EC_{Ti} = initial\ conductivity$

’ $EC_{25°C} = conductivity\ at\ 25°C$

After temperature compensation the EC value is always as it would be at $25°C$ thus eliminating any effect caused by the change in temperature.

5.5 Measuring temperature and humidity

The measurement of temperature and humidity is simplified using the SHTx.h library. The data is called using two simple functions; sht1x.readTemperatureC() and sht1x.readHumidity().

Sample code:

```
void readsht(){    //read the sht temperature and humidity values
temp_c = sht1x.readTemperatureC(); //Get temperature
humidity = sht1x.readHumidity(); }   //Get humidity
```

5.6 Controlling the motors

Running the motors is simplified using the Servo.h library. The required action is called using “motor();” function which has three parameters.

motor(3, FORWARD, 255);
First variable indicates the port on the motor shield to which the motor is attached to, in this case port 3. Second variable indicates the direction “FORWARD” or “BACK-WARD”. Third variable controls the speed of the motor (0-255).

The same function is used to stop the motor the only difference is that instead of the "FORWARD/BACKWARD" variable is changed into "RELEACE".

Sample code:

```
  motor(3, BACKWARD, 255);     //Run motor
  delay(500);                  //Wait 500ms
  motor(3, RELEASE, 0);        //Stop motor
```

5.7 Saving data to SD card

Data (Text string) is saved to the SD card using "SD.open("datalog.txt", FILE_WRITE);" function.

First the all of the data is grouped into the "dataString" variable after that variable is written into the file specified in the function (datalog.txt).

Sample code:

```
void savesd(){
 String dataString = "";
 dataString=  String(temperature) + String(humidity) + String(pH) + String(EC);
 //write data to sd card
 // open the file. note that only one file can be open at a time,
 // so you have to close this one before opening another.
 File dataFile = SD.open("datalog.txt", FILE_WRITE);
 // if the file is available, write to it:
 if (dataFile) {
   dataFile.println(dataString);
   dataFile.close();
   // print to the serial port too:
   sdcard = 1;
  }
 // if the file isn't open, pop up an error:
```

```
else {
  sdcard = 0;
 }
}
```

5.8 Displaying data on TFT screen

The Adafruit TFT library (Adafruit_GFX.h and Adafruit_ST7735.h) is used to simplify the procedure.

Sample code:

```
tft.setCursor(0, 0);                     //Set the position of the cursor
tft.fillScreen(ST7735_BLACK);            //Clear the screen with black color
tft.setTextColor(ST7735_WHITE);          //Set text color to white
tft.setTextSize(2);                      //Set font size to 2 (max3)
tft.print("Tmp1: ");                     //Display "Tmp1:" text
tft.setTextSize(3);                      //Set font size to 3
tft.setTextColor(ST7735_GREEN);          //Set text color to green
tft.println(celsius, 1);                 //Print variable "Celsius" with one decimal
```

6 TEST SETUP

The testing of the device was conducted in TAMK's greenhouse (Figure 12). The test was running for 21 days during which pH and EC was controlled only by the device. The nutrient solution was changed once on the eleventh day.

Figure 12. TAMK's greenhouse(A.Sundgren)

6.1 Top fed drip system

The top fed drip system was constructed from scrap material (Figure 13).

The main parts were:

- Aquarium
- Styrofoam
- Two pots
- Silicone hose
- 3-way junction for silicone hose
- Air pump
- Air stone
- Water pump
- Tape

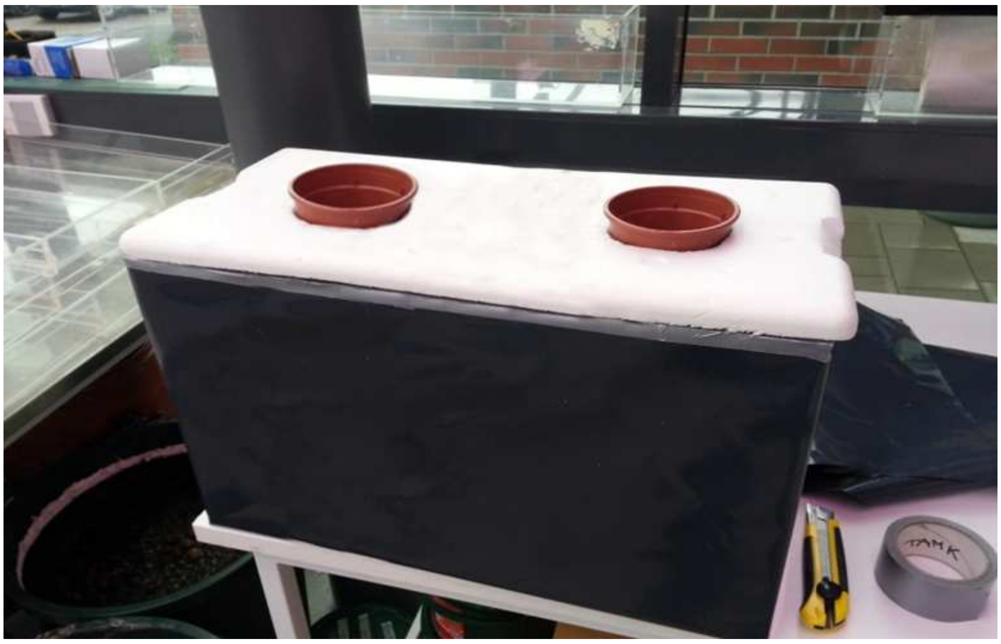


Figure 13. Top fed drip system (A.Sundgren)

6.2 Principle of operation

Nutrients needed by plants are added to a tank of water to create a nutrient reservoir.
The nutrient solution in the tank is then pumped up through the tubing to the base of each plant, the solution then flows through the root network providing easily accessible nutrients for the plants. The solution then drips back into the reservoir and the cycle starts again.
To retain good dissolved oxygen levels it is essential to aerate the water, this is achieved using air pump and air stone. The air stone is place on the bottom of the tank to provide continuous oxygenation of the nutrient solution (Figure 14).

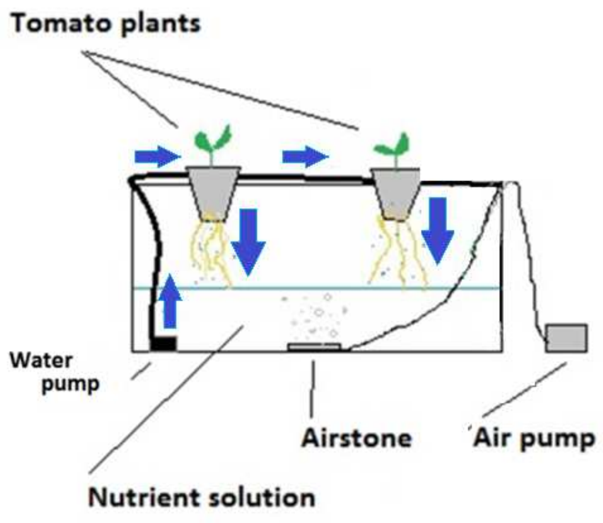


Figure 14. Circulation of nutrient solution in top drip system (A.Sundgren)

The pH of the nutrient solution tends to get higher over the time and thus needs to be lowered as needed. Around 2ml of diluted phosphoric acid is pumped into the tank using small peristaltic pump to lower the pH (Figure 15).

Constant evaporation of the water from the tank can lead to higher concentration of nutrients. The simplest way of lowering the concentration of nutrients is to add some water into the reservoir. This is accomplished using second peristaltic pump. As soon as the device detects high EC value it turns on the pump and around 2dl of water is pumped into the tank from a vessel which contains only water (Figure 15).

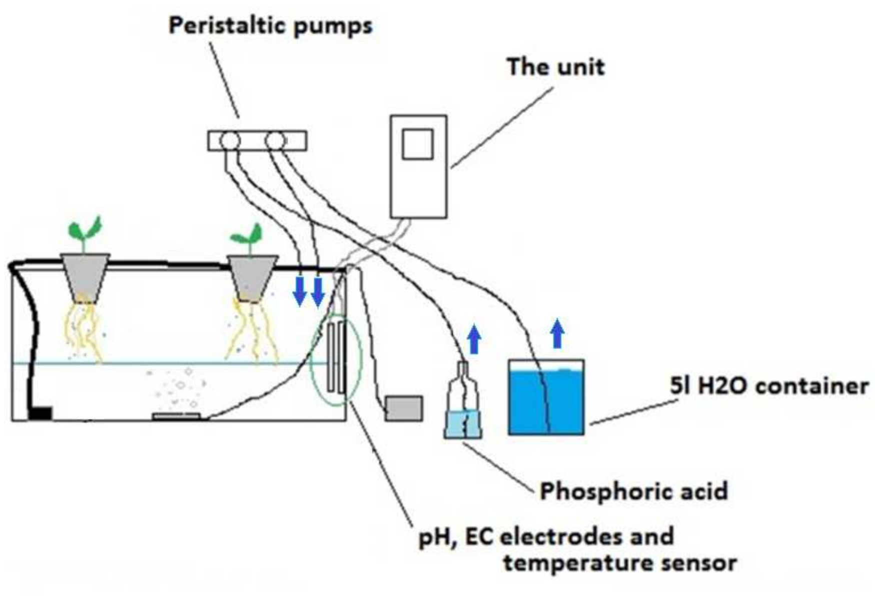


Figure 15. Description of the automation setup (A.Sundgren)

6.3 Flowchart

Figure 16 presents the flowchart which describes one program cycle during which all of the parameters are measured and if needed also manipulated.

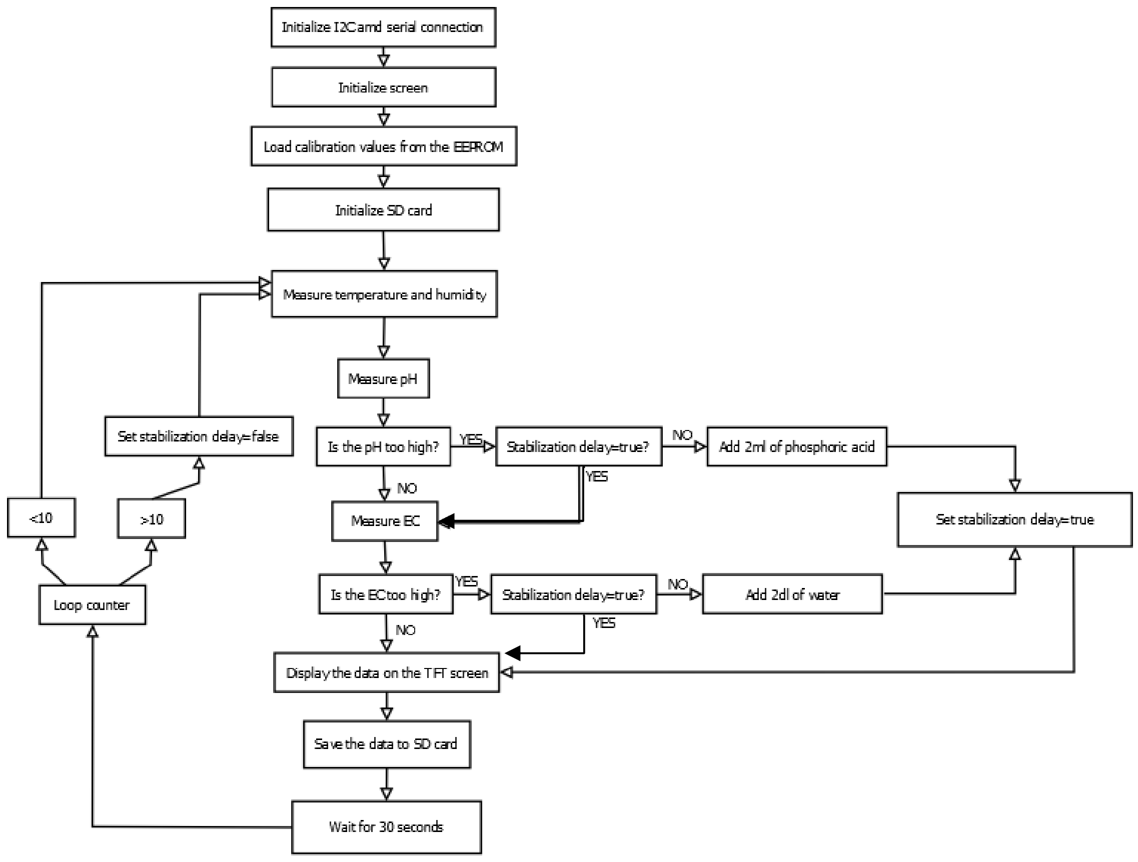

Initialize I2C and serial connection
Initialize screen
Load calibration values from the EEPROM
Initialize SD card
Measure temperature and humidity
Measure pH
Set stabilization delay=false
Is the pH too high?
YES
Stabilization delay=true?
NO
Add 2ml of phosphoric acid
YES
NO
<10
>10
Measure EC
Set stabilization delay=true
Loop counter
Is the EC too high?
YES
Stabilization delay=true?
NO
Add 2dl of water
NO
YES
Display the data on the TFT screen
Save the data to SD card
Wait for 30 seconds

Figure 16. Simplified flowchart

7 RESULTS

The unit was running for 21 days during which it was monitoring and controlling pH and EC levels fully autonomously. During that time it also measured air temperature, temperature of nutrient solution and humidity. All of the measurements were logged onto SD card for later analysis. During one day the prototype did on average around 2500 measurements, because of the missing time module determining the exact time is difficult and thus the x-axis uses days as she scale. Nerveless it is easy to determine night and day time by fluctuation of the temperature (Figure 17).
During the experiment the motor trigger level for pH and EC were adjusted.
The nutrient solution was changed to fresh one on eleventh day.

7.1 pH regulation

The initial maximum pH level (motor trigger level) was set to pH 6,6. As soon as the pH reached pH 6,6 the unit activated pump which pumped 2ml of phosphoric acid into the nutrient solution, this caused immediate drop of pH (Figure 17, vertical red stripes). After the addition of phosphoric acid the pH dropped, but then started to slowly increase until reaching once again the pH6,6, this cycle continues through whole experiment. The frequency of the zigzag pattern (pH corrections) is clearly higher after addition of the fresh nutrient solution (day1 and day 11) (Figure17). After the addition of fresh nutrient solution the motor triggering level was lowered to pH6 (day 11).

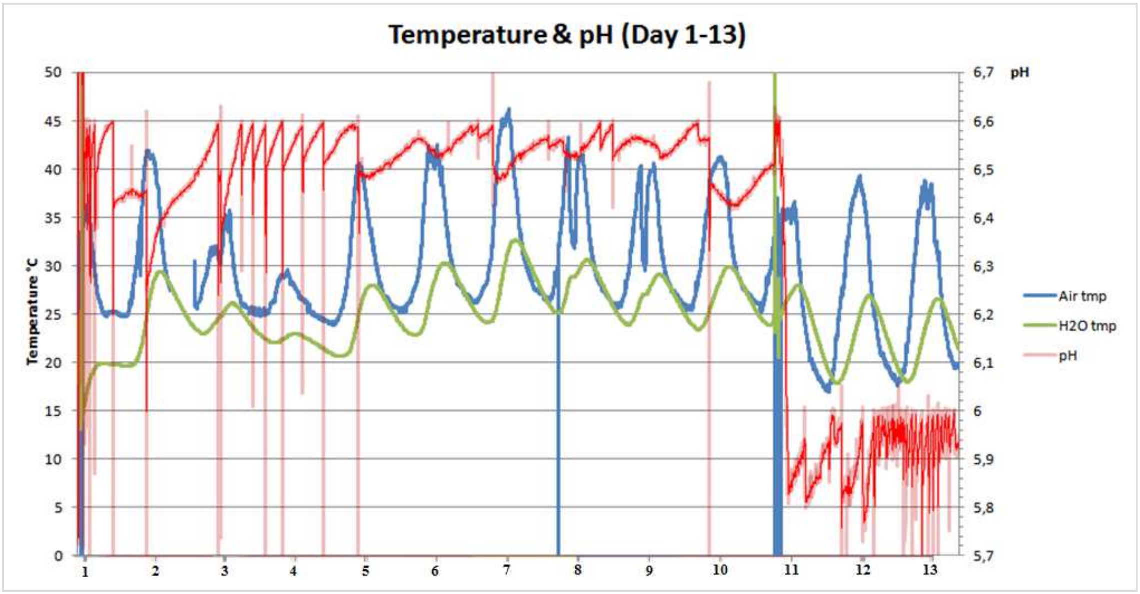


Figure 17. Temperature and pH data during day 1-13

The similar zigzag pattern continues in figure 18 (day 14-21). During the experiment the air temperature in the greenhouse was changing from 15°C to 46,2°C. The big fluctuation in the air temperature resulted on one side from poor ventilation and on another side from poor heating of the greenhouse. The temperature of the nutrient solution was fluctuating between 15°C to 33°C.

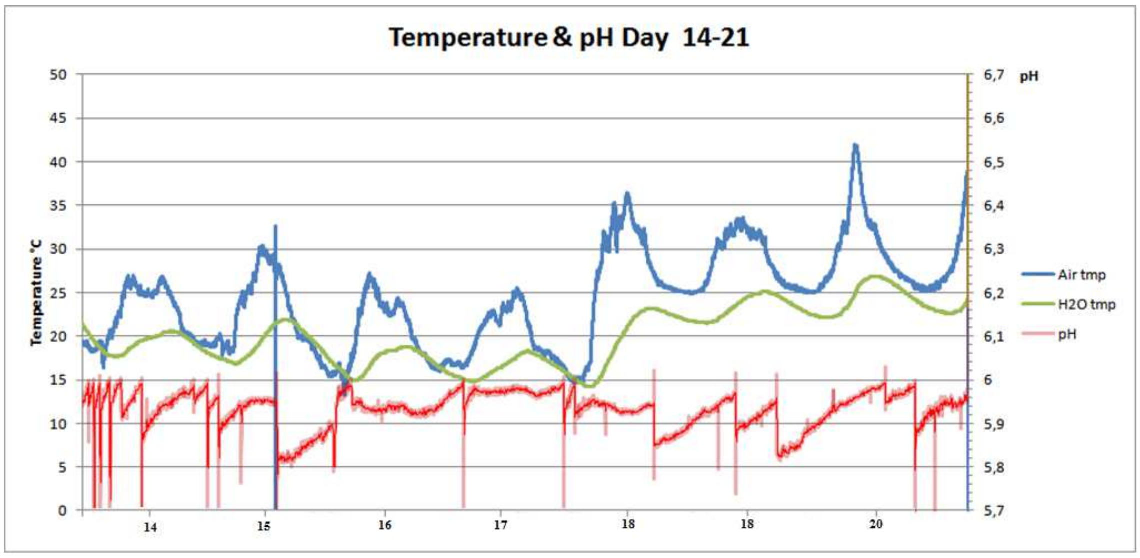


Figure 18. Temperature and pH data during day 14-21

7.2 EC regulation

The initial maximum EC level (motor trigger level) was set to 0,8mS. When EC reading reached 0,8mS the unit activated pump which pumped 2dl of tap water into the nutrient solution. (Figure 19). The addition of water didn't case substantial drop in EC value, but merely stopped it from rising. The big red area visible at day 11 (Figure 19) resulted from change of nutrient solution to fresh one.

The maximum EC level was adjusted to 0,85mS on day 8, to 1,19mS on day 11 (Figure 19), and finally to 1,3mS on day 18 (Figure 20).

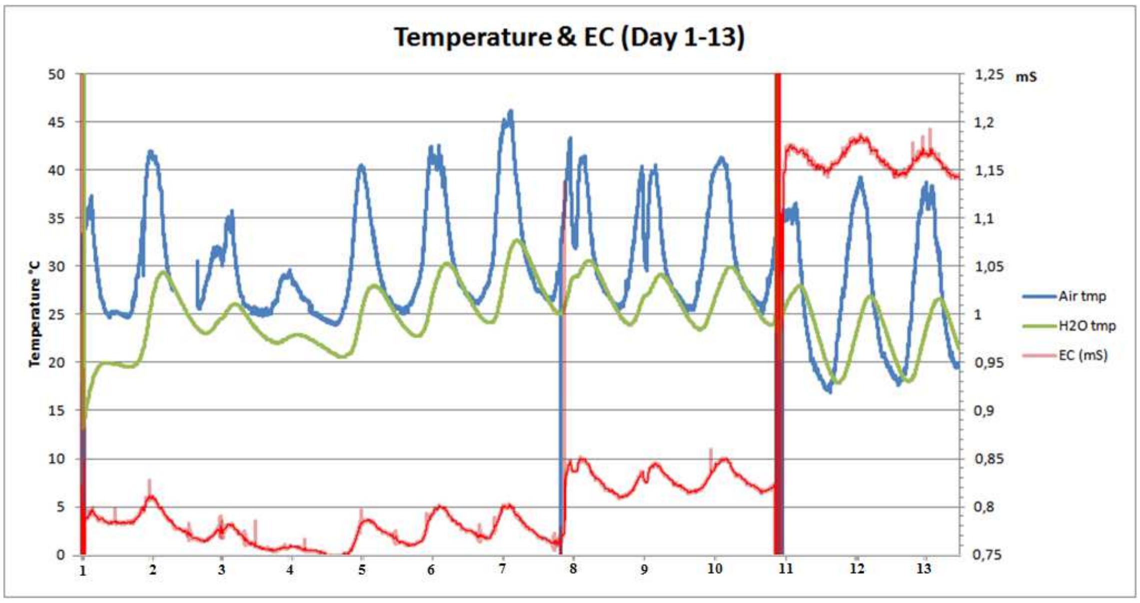


Figure 19. Temperature and conductivity (EC) data during day 1-13

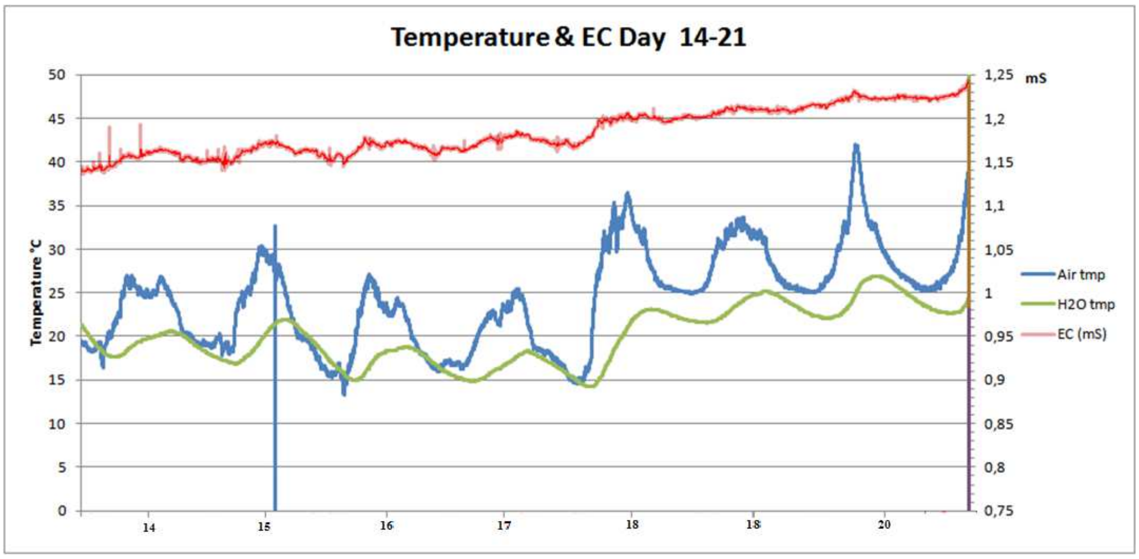


Figure 20. Temperature and conductivity (EC) data during day 14-21

The fluctuation in EC value is more obvious during the days 1-13 (Figure 19) than during the days 14-21 (Figure 20). During the cloudy days the temperature inside the greenhouse stayed around 10°C lower than during the sunny days. One such day was day 4 (Figure 19) and also days 14-16 (Figure 20). By observing the EC levels during these days it can be seen that the evaporation rate of the nutrient solution was lower as there was no substantial variation in the EC measurement data.

Figure 21 displays the tomato plant growth rate during the experiment.

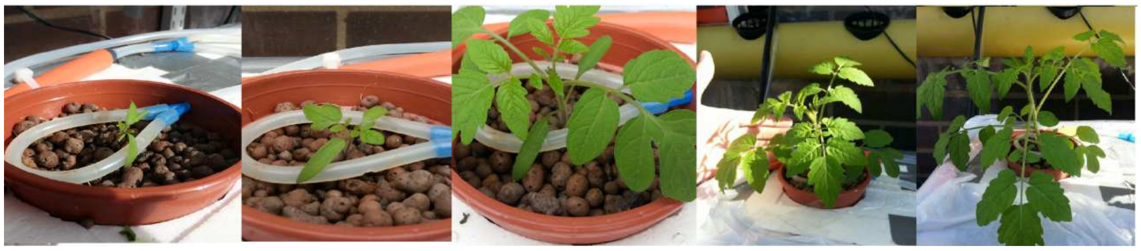


Figure 21. Tomato growth during the experiment. From left to right, Day 1, Day 4, Day 7, Day 14, day 21

8 DISCUSSION

Data recorded during the experiment suggest that the prototype was working as expected. Addition of phosphoric acid reduced the pH as expected but in some situations the decrease of the pH was bit too radical. Adjusting the speed of the pump or the amount of the acid added during one pH correction cycle could be lowered to reduce the imminent impact to the pH of the nutrient solution. This would most likely remove the sudden and radical change in the pH after addition of the phosphoric acid. Also lowering the concentration of the acid could do the trick, although this way the flask which holds the diluted acid needs to be bigger as more of it will be used to gain same effect.

Closer observation of figure 17 reveals that despite of threshold level for pH adjustment was set to pH6,6 (Day 1-10) the device did adjust the pH even at pH6,5 (Day 2). There are also some measurements which are higher than pH6.7 which should not be possible. Some of this data might be simply errors created by small air bubbles on the surface of the pH electrode, there is also a possibility that the electrode was taken out of the nutrient solution this could explain the sudden pH of 6.7.

Adjustment of electrical conductivity by addition of water had desired effect. During the sunny days (Figure 19,day 1- 13), the evaporation rate of the nutrient solution was greater than during the cloudy days (figure 20, days 14- 17) and also the fluctuation in the concentration of the nutrient solution is more observable during the sunny days. The fluctuation in EC value was only 50µS, which could be lowered by lowering the amount of water that is pumped into the nutrient solution.
There are some spikes in EC data (Figure 19 &20). This abnormality might be caused by small air bubbles attached to the surface of the electrode.

The plants grew steady during the whole experiment. During the sunny days the growth rate was more rapid than during the cloudy days. Also changing the nutrient solution to fresh one accelerated the growth rate. There was some mold growing inside of the tank, this was probably caused by high temperature (>30ºC) of the nutrient solution which accelerated the bacterial and fungal growth. Increasing the ventilation of the greenhouse could help to reduce the temperature in the greenhouse and slow down the bacterial growth. Usage of water purification methods like UV-sterilization would be necessary for a bigger system as uncontrolled bacterial and fungal growth could lead to loss of harvest and cause cost escalation. During this experiment no water purification methods were used as the nutrient solution was changed to fresh one once per week, this procedure is enough for a small size system but is impractical for bigger systems as it leads to higher costs.

9 FUTURE DEVELOPMENT

All of the functions (calibration, setting threshold levels etc) of the prototype are currently controlled using computer, buy creating stand-alone graphical user interface the prototype could be controlled without the computer, this would make it more user-friendly to use.

The code consisting of 1800 lines will be shared on GitHub so that everybody interested in the subject could build one or participate in future development.

This prototype could be also used for controlling some other systems like aquaponics, aquariums, small scale water treatment etc. Basically this prototype could be used to control any simple system where measurement and manipulation of pH and/or EC is required.

REFERENCES

Adafruit. 2014. *Waterproof DS18B20 Digital temperature sensor + extras*. [ONLINE] Available at: http://www.adafruit.com/images/970x728/381-00.jpg.
Adafruit motor/stepper/servo shield [ONLINE] Available at: http://www.adafruit.com/product/81. *MicroSD card breakout board* [ONLINE] *http://www.adafruit.com/product/254* [Accessed 02 February 15].

Atmel Corporation(2014). *Atmel ATmega640/V-1280/V -1281/V-2560/V-2561/V.* [ONLINE]. Available at: *http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_Summary.pdf*
[Accessed: 26.04.2014].

Arduino.cc(2014), *Arduino Mega 2560 board* [ONLINE]. Available at: http://arduino.cc/en/uploads/Main/ArduinoMega2560_R3_Fronte.jpg [Accessed 26.04.2014]. & *http://arduino.cc/en/Main/arduinoBoardMega* [Accessed: 26.04.2014].

Jan Borchers(2013). Arduino in a Nutshell. [ONLINE]. Available at: hci.rwth-aachen.de/arduino [Accessed: 11.05.2014].

PASCO(2001). *pH Temperature Compensation.* [ONLINE]. Available: http://www.pasco.com/support/technicalsupport/technote/techIDlookup.cfm?TechNoteID=281 [Accessed: 11.05.2014].

Practicalmaker (2014) *Documentation for the EC Shield.* [ONLINE]. Available at: *http://www.practicalmaker.com/documentation/ec-shield-documentation*
[Accessed: 21.05.2014].

Sensirion, (2011), *SHT11* [ONLINE]. Available at: http://www.sensirion.com/typo3temp/pics/13519f94b7.jpg [Accessed 09 February 15].

Sparkyswidgets(2013).*MinipH-I2C-pH-Interface*.[ONLINE].Availableat: http://www.sparkyswidgets.com/portfolio-item/miniph-i2c-ph-interface
[Accessed: 11.05.2014].

Sparkys,(2014),*mini-pH-interface*[ONLINE].Available_at: http://www.sparkyswidgets.com/wp-content/uploads/MinipH_inaction_med.png
[Accessed 17].

T.Asao(2012). *Hydroponics - A Standard Methodology for Plant Biological Re-searches*. 1ST Edition. Intech. SBN 978-953-51-0386-8, Rijeka, Croatia

M.Mäntynen(2001) *Temperature correction coefficients of electrical conductivity and of density measurements for saline groundwater.* [ONLINE]. Available at: *http://www.posiva.fi/files/2094/POSIVA-2001-15_Working-report_web.pdf*[Accessed: 21.05.2014].

Marschner, H.(1995). Mineral Nutrition of Higher Plants, Academic Press, ISBN 0-12-473542-8, New York, U. S. A.