VIET NAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

ELECTRICAL – ELECTRONIC DEPARTMENT

NINH THE VINH CUONG

APPLYING IOT INTO SEAFOOD-PROCESSING WASTEWATER TREATMENT

INDIVIDUAL PROJECT

ELECTRICAL ENGINEERING

HO CHI MINH CITY. 2021

VIET NAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

ELECTRICAL – ELECTRONIC DEPARTMENT

NINH THE VINH CUONG – 418H0363

APPLYING IOT IN SEAFOOD-PROCESSING WASTEWATER TREATMENT

INDIVIDUAL PROJECT

ELECTRICAL ENGINEERING

Instructor

PhD. Dinh Hoang Bach

HO CHI MINH CITY, 2021

ACKNOWLEDGE

I sincerely thank Dr. Bach H. D. - Lecturer of Ton Duc Thang University. Dr.Bach directly instructed and helped me to implement and complete thiss individual project. In the process of learning and implementing the project, although many efforts have been made, it is still impossible to avoid shortcomings, large or small. However, in return I have received the dedicated guidance of Dr.Bach helped me overcome the mistakes I often make and encounter in my studies. Dr.Bach was also the one who directly taught me the subject of "Industrial Communication Networks". This project is the result of my study and practice during nearly 7 semesters at Ton Duc Thang University. Therefore, I also sincerely thank all the teachers and faculty of Electrical - Electronic Engineering of Ton Duc Thang University - who participated in teaching and equipped me with the knowledge to help me complete. This project.

Next, I also sincerely thank my family and friends who have always been by my side and encouraged me during this project as well as throughout the study process.

Sincerely.

Ho Chi Minh City, 25th December 2021

Author

[illegible]

Ninh The Vinh Cuong

This thesis was carried out at Ton Duc Thang University.

Scientific advisor: D r . D i n h H o a n g B a c h

This thesis is defended at the specialized projects was hold

at Ton Duc Thang University on … /…/……

Confirmation of the Chairman of the Undergraduate Thesis Examination Committee and the Dean of the faculty after receiving the modified thesis (if any).

CHAIRMAN	**DEAN OF FACULTY**
…………………………….	………………………………

WORK IS COMPLETED AT
TON DUC THONG UNIVERSITY

I hereby declare that this is my own research work and is under the scientific guidance of Dr. Bach H.D. The research contents and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the author himself from different sources, clearly stated in the reference section.

In addition, the Individual Project also uses a number of comments, assessments as well as data of other authors, other agencies and organizations, with citations and source annotations.

If there is any fraud, I would like to take full responsibility for the content of my Individual Project. Ton Duc Thang University is not related to copyright and copyright violations caused by me during the implementation process (if any).

Ho Chi Minh City, 25th December 2021

Author

[illegible]

Ninh The Vinh Cuong

TRƯỜNG ĐH TÔN ĐỨC THẮNG
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ ĐỒ ÁN CHUYÊN NGÀNH

(*Ghi chú: Bảng nhiệm vụ này đóng vào trang thứ nhất của đồ án*)

Họ và tên: Ninh Thế Vĩnh Cường	MSSV: 418H0363
Ngành: Kỹ Thuật Điện	Lớp: 18H40101

A. Đề tài: Áp dụng IoT vào quy trình xử lý nước thải của ngành chế biến thủy sản.

B. Nhiệm vụ đề tài:

- Hiểu rõ đề tài
- Tìm kiếm các dự án có sẵn liên quan với đề tài
- Tìm kiếm các thiết bị phần cứng, phần mềm phù hợp với đề tài
- Biết cách kết nối phần cứng, sử dụng phần mềm
- Biết cách gửi dữ liệu lên điện toán đám mây
- Làm mô hình
- Đánh giá kết quả mô hình

C. Ngày giao đồ án: 20/09/2021

D. Ngày nộp đồ án: 24/12/2021

E. Ngày bảo vệ trước hội đồng: 30/12/2021

F. Họ và tên giảng viên hướng dẫn: **TS. Đinh Hoàng Bách**

TP HCM, ngày 20 tháng 09 năm 2021

CNBM. Kỹ Thuật Điện	Giảng viên hướng dẫn
[illegible]	[illegible]
TS. Đinh Hoàng Bách	**TS. Đinh Hoàng Bách**

TON DUC THANG UNIVERSITY **ELECTRICAL – ELECTRONIC** -------------------	SOCIALIST REPUBLIC OF VIETNAM Independence - Freedom – Happiness ----------------------

INDIVIDUAL PROJECT SCHEDULE

Student: Ninh The Vinh Cuong
Class: 18H40101 ID: 418H0363
Project: Applying IoT to seafood processing wastewater treatment

Week/Day	**Work**		**Instructor**
	Done	**Continue**	
Week 1 (02/10/2021 - 09/10/2021)	- Meet with the instructor to listen to the requirements of the project, proceed to select the project. - The instructor conducts the review of the topic	- Meet with the instructor to listen to the requirements of the project, proceed to select the project. - The instructor conducts the review of the topic	[illegible]
Week 2 (09/10/2021 - 16/10/2021)	- Found topic for project - Reading basics of IoT	- Studying about basics of IoT - Start to develope the solutions	[illegible]
Week 3 (16/10/2021 – 23/10/2021)	- Have the basic concept of solution	- Stuying about basics of IoT - Finding equipmients	[illegible]
Week 4 (23/10/2021 – 30/10/2021)	- Learned about basics of IoT - Started to oder some equipments	- Learning about Arduino - Learning how to wire, design circuit	[illegible]

Week 5 (30/10/2021 – 06/10/2021)	- Try to have make a basic circuit	- Learning more about modules, sensors	[illegible]
Mid-term check	Evaluation of completed volume ……..% be allowed to continue/discontinue Individual Project.		
Week 6 (06/10/2021 – 13/11/2021)	- Learned how to connect pins of modules - Started to write report	- Learning how to code in Arduino IDE, libraries. - Writing report	[illegible]
Week 7 (13/11/2021 – 20/11/2021)	- Learn how to code/upload code to arduino - Writed part of report	- Learning how to send data to Thingspeak through API - Writing report	[illegible]
Week 8 (20/11/2021 – 27/11/2021)	- Learned how to send data to Thingspeak. - Writed part of report	- Finishing model and make some samples	[illegible]
Week 9 (27/11/2021 – 04/12/2021)	- Finish the model and test some samples	- Finishing the report	[illegible]
Week 10 (04/12/2021 – 11/12/2021)	- Finish the report		[illegible]
Submit	Completed……..% Individual Project protected/unprotected Individual Project.		

APPLYING IOT INTO SEAFOOD-PROCESSING WASTEWATER TREATMENT

ABSTRACT

This project working on seafood-processing wastewater treatment system. It wouldd like to take part in the system and reduce the time, as well as the cost of chemicals use in the process. Using sensors to collect information of the amount of contaminats in the wastewater and applying IoT to send it the cloudbase server. The data are stored and will be supervised by operators for decisions. In the future, Machine Learning or Artificial Intelligent might be take part in the process.

TABLE OF CONTENTS

TABLE OF CONTENTS IX

LIST OF FIGURES XI

LIST OF TABLES XIII

LIST OF ABBREVIATIONS XIV

CHAPTER 1.INTRODUCTION 15

1.1 INTRODUCTION 15
1.2 REQUIREMENTS 16
1.3 IDEA AND IMPLEMENTATION METHOD 16
1.4 CURRENT SOLUTIONS 17

CHAPTER 2.THEORIES 18

2.1 COMMUNICATION BETWEEN MODULES AND CONTROLLERS 18
2.1.1 UART 18
2.1.2 SPI 19
2.1.3 I2C 21
2.1.4 1-Wire 24
2.2 COMMUNICATION WITH THE INTERNET 25

CHAPTER 3.IMPLEMENTATION SOLUTION 26

3.1 VIET NAM STANDARD ON CONTAMINANTS IN WASTEWATER 26
3.2 CURRENT WASTEWATER TREATMENT PROCESS 28
3.3 APPLY THIS PROJECT INTO THE CURRENT PROCESS 32

CHAPTER 4.HARDWARE AND SOFTWARE 36

4.1 SOFTWARE 36
4.1.1 Arduino IDE 36
4.1.2 ThingSpeak 37
4.2 HARDWARE 38

4.2.1 Sensors *38*
4.2.2 Display *44*
4.2.3 Controllers *45*
4.2.4 Module for connection between controllers *48*
4.2.5 Module for connection between controller and Internet *49*
4.2.6 Power Supply *50*

CHAPTER 5.MODEL CONSTRUCTION AND RESULTS 51

5.1 MODEL CONSTRUCTION 51
5.2 RESULTS 53
5.2.1 Gains *56*
5.2.2 Obstacles *56*

CHAPTER 6.CONCLUSIONS 57

6.1 CONCLUSIONS 57
6.2 FUTURE DEVELOPMENTS 57

REFERENCES 58

APPENDIX 1

LIST OF FIGURES

Figure 1.1 Seafood production lines *(Source: Internet)* 15
Figure 1.2 Functional block diagram highlighting core components of analytical system. [3] 17
Figure 1.3 Design of the device. [3]..17
Figure 2.1 UART Communication..18
Figure 2.2 UART’s Dataframe..19
Figure 2.3 SPI Communication..20
Figure 2.4 Masters and Slaves connections in SPI 21
Figure 2.5 I2C Communication..22
Figure 2.6 I2C's dataframe 23
Figure 2.7 I2C one Master, many Slaves; and many Mastes many Slaves connections 23
Figure 2.8 1-Wire bus communication 24
Figure 2.9 1-Wire transmission basis..25
Figure 2.10 WiFi general frame format 25
Figure 3.1 Current wastewater treatment process..28
Figure 3.2 Applying this project into current seafood processing wastewater treatment 35
Figure 4.1 Arduino Logo..36
Figure 4.2 Arduino IDE interface 36
Figure 4.3 ThingSpeak..37
Figure 4.4 Optical DO sensor..38
Figure 4.5 Polarographic DO sensors 39
Figure 4.6 Galvantic DO sensor..39
Figure 4.7 Dissolved Oxygen sensor 40
Figure 4.8 How pH sensor work 41
Fig ure 4.9 DFRobot’s Analog pH sensor..42

Figure 4.10 Turbiduty sensor 43
Figure 4.11 Temperature sensor..43
Figure 4.12 TDS sensor...44
Figure 4.13 0.96 inch OLED screen 44
Figure 4.14 Seeeduino's PINs 48
Figure 4.15 NRF24L01 + PA + LNA module 49
Figure 4.16 ESP-01S..49
Figure 4.17 ESP-01/ESP-01S pinout 50
Figure 4.18 AC-DC Power Supply 3.3VDC...50
Figure 5.1 Data flow in this project 51
Figure 5.2 Wiring for the project 52
Figure 5.3 Sender (*left)* and Receiver (*right*). 53
Figure 5.4 Samples (Milk, Tea, Cooking oil, Salt water, Tap water).......................53
Figure 5.5 OLED screen when Sender (*left*) and Receiver (*right*) are working.......54
Figure 5.6 Measuring (tea sample)...54
Figure 5.7 Serial monitor of Arduino IDE for debugging when working................55
Figure 5.8 My channel to store data (1) 55
Figure 5.9 My channel to store data (2) 56

LIST OF TABLES

Table 2.1 Some specifications of UART 18
Table 2.2 Some specifications of SPI 20
Table 2.3 Some specifications of I2C 22
Table 3.1 Polluting substances in wastewater....................27
Table 3.2 Detecting techniques for some wastewater parameters 27
Table 3.3 The value of Kw and pH with increasing temperature 33
Table 3.4 Typical pH values for solutions at different temperatures....................33
Table 4.1 Seeeduino's specifications....................47
Table B.1 Arduino and TDS+Temp sensor pins connections....................5
Table B.2 Arduino and NRF24L1 pins connections....................5
Table B.3 Arduino and OLED screen pins connections 5
Table B.4 Arduino and ESP8266 pins connections 6

LIST OF ABBREVIATIONS

IoT	Internet of Things
LED	Light Emitted Diode
IDE	Integrated Development Environment
UART	Universal asynchronous receiver-transmitter
SPI	Serial Peripheral Interface
I2C	Inter-Integrated Circuit
LCD	Liquid-crystal display
UASB	Upflow Anaerobic Sludge Blanket
COD	Chemical Oxygen Demand
BOD	Biological Oxygen Demand
TSS	Total suspended solids
TDS	Total dissolved solids
DO	Dissolved Oxygen
MLSS/L	Mixed liquor suspended solids per litre
MLVSS.day	Mixed liquor volatile suspended solids times day
F/M	Food to Mass/ Food to Microorganism
SWD	Serial Wire Debug
DAC	Digital-to-Analog Converter
GPIO	General Purpose Input/Output
PIN	Process Identification Number
API	Application Programming Interface
PWM	Pulse-Width modulation
SRAM	Static random-access memory
ATC	Automatic Temperature Compensation
STC	Solution Temperature Compensation
TTL	Transistor-Transistor Logic
VCC	Voltage Common Collector
EC meter	Electrical charge meter
OLED	Organic Light Emitting Diode

CHAPTER 1. INTRODUCTION

1.1 Introduction

Environmental issues are a matter of concern these days. With the rise of pollution levels and depletion of the ozone layer, all the countries in the world are concerned about the environment and Viet Nam is not an exception.

Figure 1.1 Seafood production lines *(Source: Internet)*

Water pollution occurs when bad chemicals or microorganisms are released into water, which results life quality of human and animal being degrading.
In Viet Nam, seafood enterprises will be classified as a high risk of pollution, which mean they will need to reduce the discharge volume of wastewater into the environment, 5 times to be specific (Phương, 2021). That is why when exceeding the allowable wastewater flow, companies need to install automatic monitoring system.

The major aim of wastewater treatment is to remove as much of the suspended solids as possible before the remaining water, called effluent, is discharged back to the environment. The quality of treated wastewater is defined by physical-chemical parameters such as pH, temperature, conductivity, turbidity, Biological Oxygen Demand (BOD), Chemical Oxygen Demand (COD), Total Organic Carbon (TOC), Total Suspended Solids (TSS), and nitrogen and phosphorus compounds (Thomas, O.; Théraulaz, F.; Cerdà, V.; Constant, D.; Quevauviller P, 1997). As solid material decays, it uses up oxygen, which is needed by the plants and animals living in the water.

This project will be installed as a part of the system, which will track the amounts of polluting substances in water. Then data will be sent to server and showing it on the website. The person who has the control of this system then can decide the amount of chemicals will be used to treat this wastewater. This process help reduce the chemicals will be used and the time needed to supply oxygen in this system to minimal.

1.2 Requirements

For this project to be able to succeed, there are a few requirements:

- This project has to be able to track and alert the amount of chemicals in the wastewater, which will be sent to a website.
- Apply Internet of Things into this project, send data to the internet and display it for everyone.

1.3 Idea and implementation method

- Using Viet Nam Standard to determine wastewater chemicals.
- Learning about seafood productions' wastewater treatment process.
- Using sensors to detect the amount of chemicals, dissolved oxygen, etc. in the water.
- Apply this project into the current treatment.
- Receive data from sensors and send it to cloud for supervisory.

1.4 Current solutions

Present solutions are existed in the world, here is how one of them work. The sample intake system is comprised of a 12 V high flow pump and a reservoir. The pump draws sample from the water source filling the reservoir. The sample intake pump runs at the beginning of each analysis cycle for 30 s. The pump module is responsible for loading sample into the system and pumping the eluent through the ion exchange column and detector for analysis. The below figures show the functional block diagram and design of the device (Martínez, R., Vela, N., el Aatik, A., Murray, E., Roche, P., & Navarro, J. M., 2020).

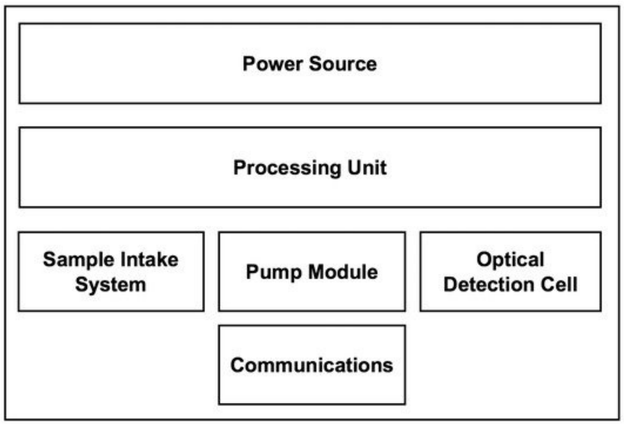


Figure 1.2 Functional block diagram highlighting core components of analytical system. [3]

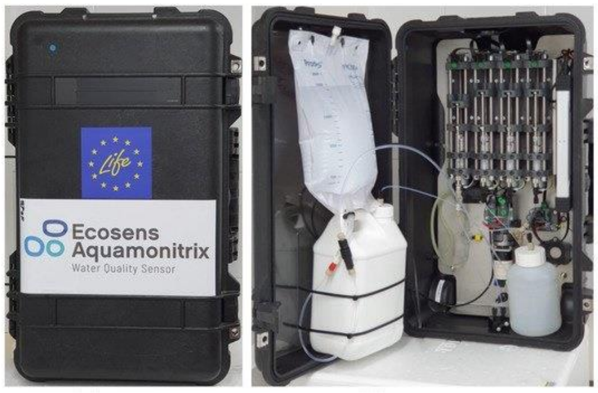


Figure 1.3 Design of the device. [3]

As it is showed, the model needs to take sample into a resevoir with a pump system. One of the aims of this project is to remove the sample-taken part of the system.

CHAPTER 2. THEORIES

This project use sensors with various kinds of communication, which is why these theories need to be mentioned.

2.1 Communication between modules and controllers

2.1.1 UART

Universal asynchronous receiver-transmitter is a devive-to-device communication protocol. UART is a serial protocol, which mean data being transmitted bit by bit using a single wire. Two wires are needed for successful serial data transfer for using 2-way communication. The cost of implementation is reduced byusing serial communication since it requires kess circuitry and wires.

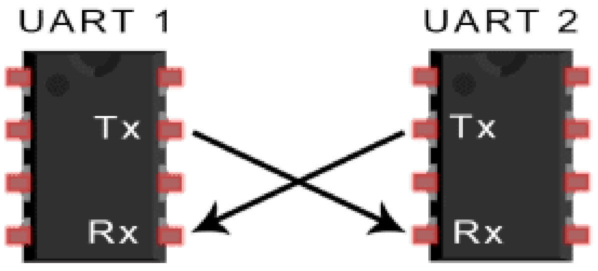


Figure 2.1 UART Communication

Signals of each UART device are called:

- Transmitter (Tx)
- Receiver (Rx)

Table 2.1 Some specifications of UART

Wires	2
Speed	9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600, 1000000, 1500000
Transmission’s method	Asynchronous

Serial or Parallel?	Serial
Maximum Masters	1
Maximum Slaves	1

Since asynchronous using packets to deliver data, UART is also the same.

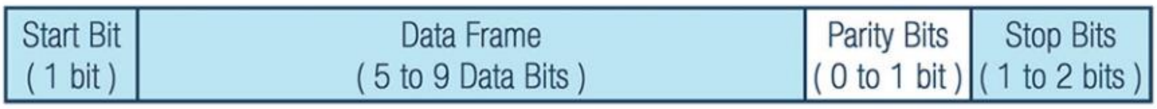


Figure 2.2 UART's Dataframe

ADVANTAGES

- Two wires only.
- The need for clock signal is unnecessary.
- Error checking with parity bit.
- If both sides are set up for special structure, it can be use
- Common.

DISADVANTAGES

- Limited dataframe's size (9 bits).
- Multiple slave or multiple master systems are unsupported.
- The baud rates of each UART must be within 10% of each other

2.1.2 SPI

Serial Peripheral Interface is a communication protocol. One unique benefit of SPI is the fact that any number of bits can be sent or received in a continuous stream while with I2C and UART, data is limited to a specific number of bits.

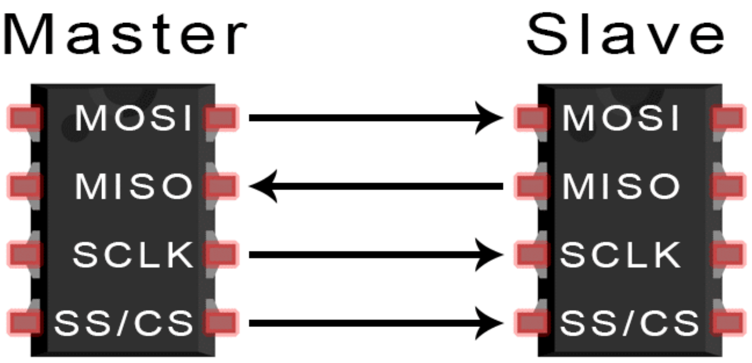


Figure 2.3 SPI Communication

Master-slave is how device communicating via SPI. The master is the controlling device, while the slave takes instruction from the master. Single master, single slave system is the easiest configuration of SPI, but one master can control more than one slave.

Signals using SPI transfer through:

- MOSI (Master Output/Slave Input)
- MISO (Master Input/Slave Output)
- SCLK (Clock) – Line for the clock signal.
- SS/CS (Slave Select/Chip Select) – Line for the master to select which slave to send data to.

Table 2.2 Some specifications of SPI

Wires	4
Speed	Up to 10 Mbps
Transmission’s method	Synchronous
Serial or Parallel?	Serial
Maximum Masters	1
Maximum Slaves	Theoretically unlimited[1]

[1] In practice, the number of slaves is limited by the load capacitance of the system, which reduces the ability of the master to accurately switch between voltage levels.

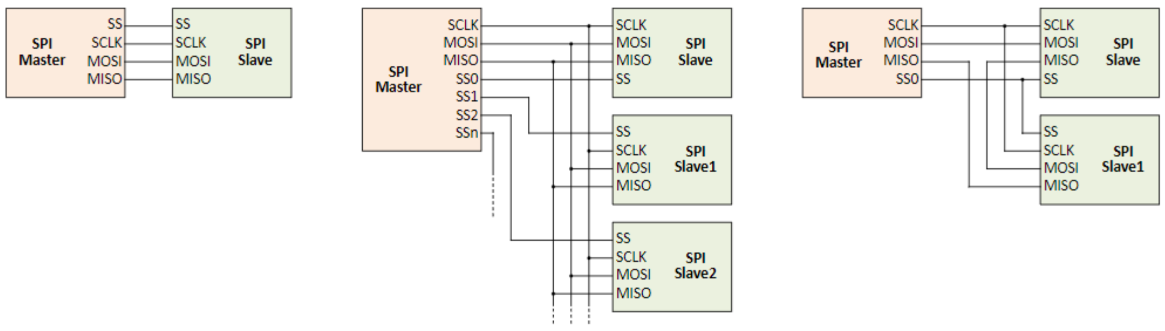


Figure 2.4 Masters and Slaves connections in SPI

ADVANTAGES

- Data can be streamed without interruption.
- No complicated slave addressing system like I2C.
- Higher data transfer rate.
- Data can be sent and received at the same time.

DISADVANTAGES

- Four wires
- Only allows for a single master.
- No acknowledgement that the data has been successfully received (I2C has this).
- No error checking.

2.1.3 I2C

Inter-Integrated Circuit is a synchronous, multi-controller/multi-target, packet switched, single-ended, serial communication bus. It is common for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication.

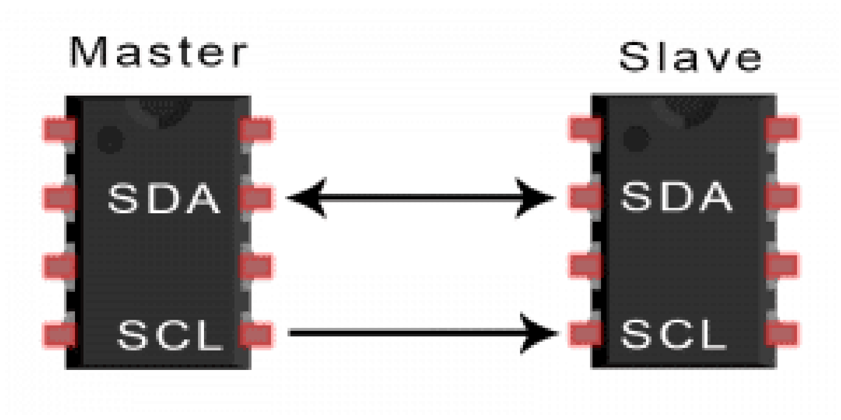


Figure 2.5 I2C Communication

I2C combines the best features of SPI and UARTs. With I2C, multiple masters control single/multiple slaves or single master connect to single/multiple slaves are all possible. This is useful when you want to have more than one microcontroller logging data to a single memory card or displaying text to a single LCD.

- SDA (Serial Data).
- SCL (Serial Clock).

Table 2.3 Some specifications of I2C

Wires	2
Speed	Standart mode: 100kbps Fast mode: 400 kbps High speed mode: 3.4 Mbps Ultra fast mode: 5 Mbps
Transmission's method	Synchronous
Serial or Parallel?	Serial
Maximum Number of Masters	Unlimited
Maximum Number of Slaves	128 (7 bits address), 1024 (10 bits address)

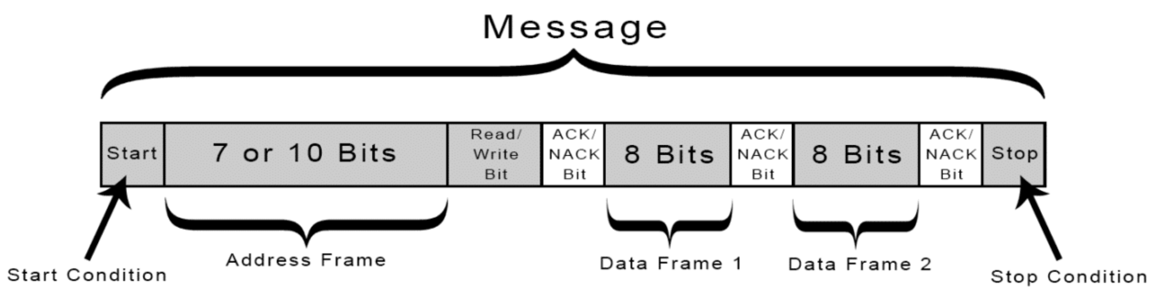
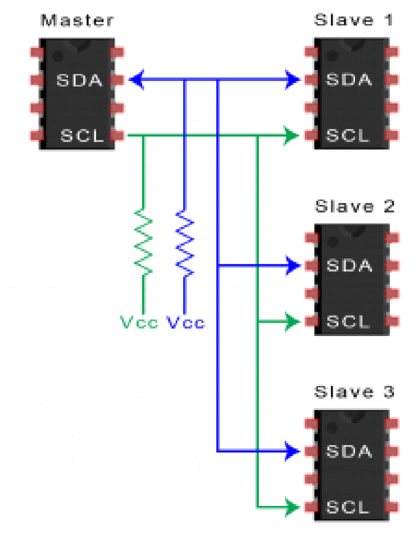


Figure 2.6 I2C's dataframe



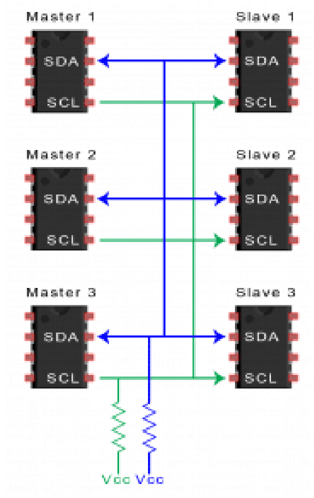


Figure 2.7 I2C one Master, many Slaves; and many Mastes many Slaves connections

ADVANTAGES

- Two wires only.
- Multiple masters and multiple slaves are supported.
- When each frame transferred successfully. There is ACK/NACK bit that gives confirmation.
- Hardware is simplier (than UARTs).
- Widely used and well know protocol.

DISADVANTAGES

- Slower data transfer rate.
- The size of the data frame is limited (8 bits).
- More complicated hardware needed.

2.1.4 1-Wire

1-Wrire uses a wire to transmit and receive, so it has a low speed. Mainly used for data collection, weather data transmission, temperature, work that does not require high speed.

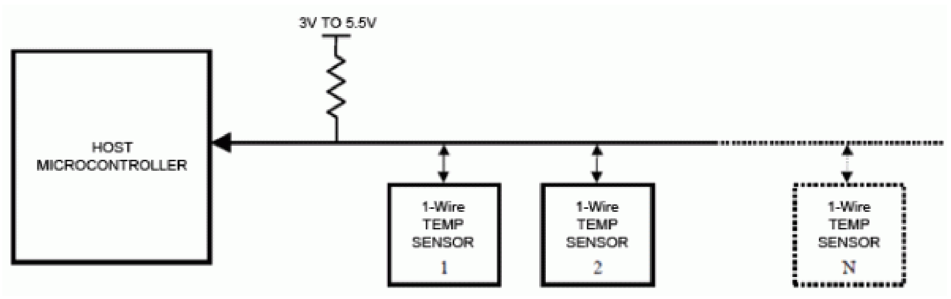


Figure 2.8 1-Wire bus communication

Transmission basis:

- The signals use Restart, 0 write, 1 write, Read.
- Write 1: (transmit bit 1) Master pulls down to 0 an interval A(us) and then back to level 1 about B.
- Write 0: (transmit bit 0) Master pulls down 0 interval C and then returns 1 interval LOW.
- Read: (read a Bit) Master pulls down to 0 for about A then returns 1. Delay about E and then read the slave value sent back. Delay F.
- Reset: (prepare to communicate) Master pulls 0 down to 0 for an H interval, then releases it to level 1, then configures the Master to be the In delay I (us) pin and then reads the returned slave value. If =0, communication is allowed. =1 transmission error or slave is busy.

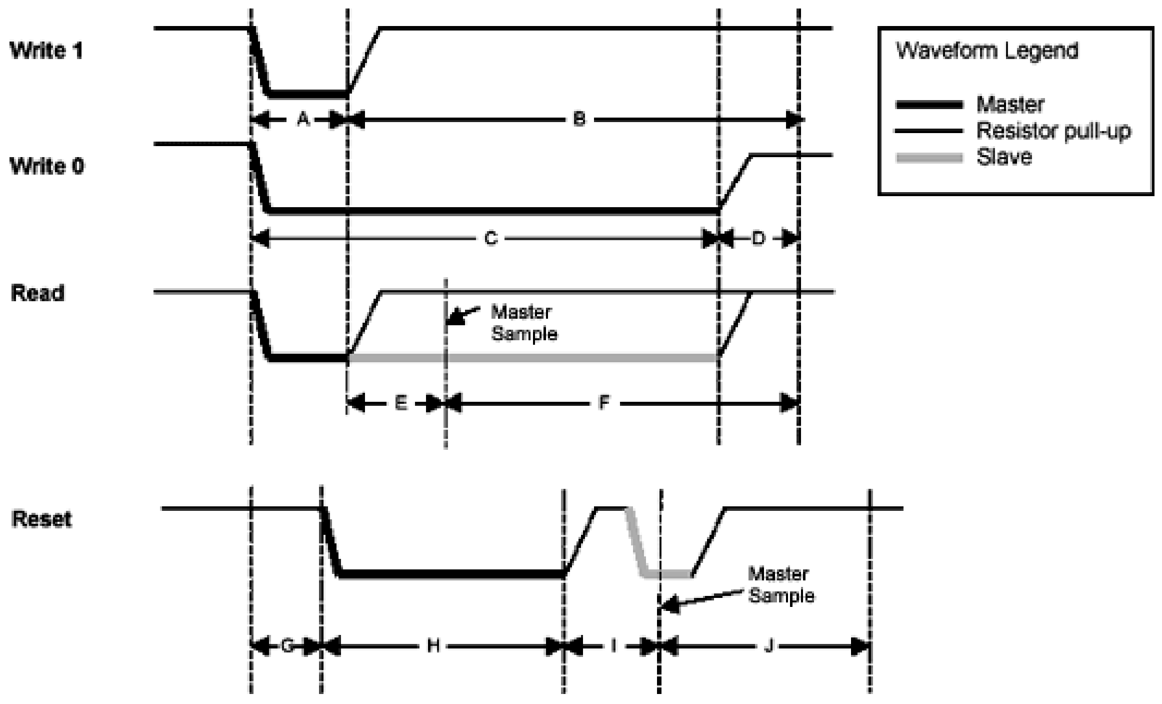
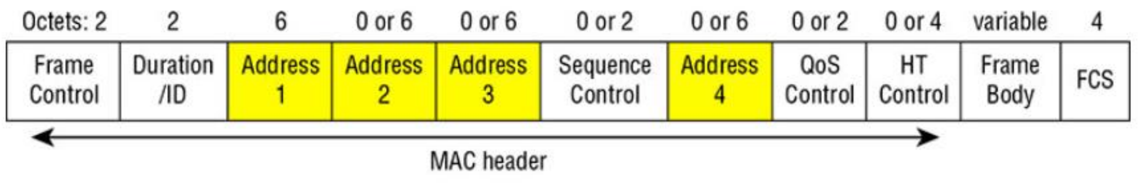


Figure 2.9 1-Wire transmission basis

2.2 Communication with the internet

WiFi works in the way other wireless devices work, uses radio frequencies to deliver signals between devices. Note that the radio frequencies are completely different from car radios, walkie-talkies, cell phones, and weather radios. It provides wireless high-speed internet and network connections by using radio waves. Wi-Fi is a trademarked phrase, and it means IEEE 802.11x.

Using an antenna, a computer's wireless adapter translates data into a radio signal and easily transmits it. After that, the signals are received and being decoded by a wireless router. Wired Ethernet connection is used by the router to send information to the internet.



Octets: 2	2	6	0 or 6	0 or 6	0 or 2	0 or 6	0 or 2	0 or 4	variable	4
Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	QoS Control	HT Control	Frame Body	FCS

MAC header

Figure 2.10 WiFi general frame format

CHAPTER 3. IMPLEMENTATION SOLUTION

3.1 Viet Nam Standard on contaminants in wastewater

As in circulars of Viet Nam Ministry of Natural Resources and Environment [1], pollution caused by wastewater at seafood processing establishments includes production wastewater and domestic wastewater:

- Production wastewater: generated during processing and cleaning water of factories, machinerry and equipments, etc. Wastewater composition contains organic substances, suspended solids, residues, microorganisms. materials and grease. The flow and composition of seafood processing wastewater varies greatly between factories depending on the source of raw materials used, and the composition of substances used in processing (detergents, additives, ...).
- Domestic wastewater: generated in toilets and canteens. Wastewater composition contains residues, suspended solids, organic substances, nutrient and microorganisms.

The maximum allowable value of pollutant parameters in seafood processing wastewater when discharged into the receiving water source is calculated according to the following formula:

$$C_{max} = C \times K_q \times K_f$$

Where:

- C_{max} is the maximum allowable value of pollutant parameters in seafood processing wastewater when discharged into the receiving water source.
- C is the value of pollutant parameters in seafood processing wastewater.
- K_q is the coefficient of wastewater receiving source.
- K_f is the discharge flow factor.

Table 3.1 Polluting substances in wastewater

No.	Parameters	Unit	C value	
			A	B
1	pH	-	6-9	5.5-9
2	BOD5	mg/l	30	50
3	COD	mg/l	75	150
4	TSS	mg/l	50	100
5	Amoni	mg/l	10	20
6	Nitro	mg/l	30	60
7	Phosphour	mg/l	10	20
8	Total animal and vegetable fats and oils	mg/l	10	20
9	Residual chlorine	mg/l	1	2
10	Total Coliforms	MPN or CFU/100ml	3000	5000

Column A Table 3.1 specifies C values of pollution parameters in aquatic product processing wastewater when discharged into water sources used for domestic water supply purposes.

Column B Table 3.1 specifies the C value of pollution parameters in aquatic product processing wastewater when discharged into water sources **not** used for domestic water supply purposes.

As you can see, 10 parameters above is where this project need to be able to observe. In these days, only some of chemicals are detectable using electronic sensors. This is a obstacle of this project, and hopefully will be solved in the future.

Table 3.2 Detecting techniques for some wastewater parameters

No.	Parameters	Detecting technique
1	pH	Sensor
2	BOD	Using DO value to calculate
3	TSS	Sensor
4	DO	Sensor

3.2 Current wastewater treatment process

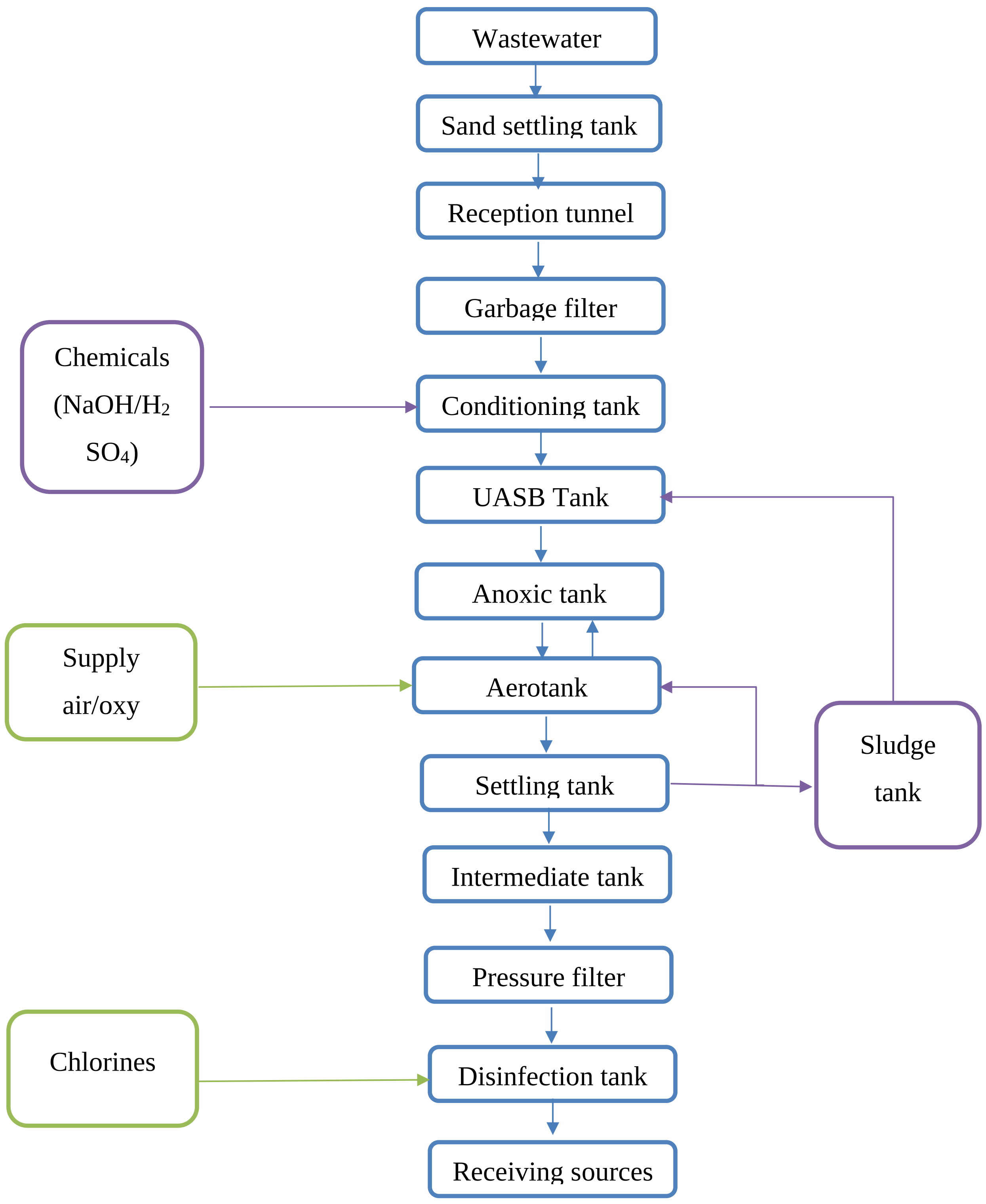


Figure 3.1 Current wastewater treatment process

Wastewater from the production workshops follows the company's channel through the coarse screen to the sand settling tank, which is placed deep underground, where it will retain the sand and large-sized suspended solids to ensure the stable operation of subsequent processing works. Before entering the sand settling tank, wastewater is led through a coarse filter device to remove large solids such as paper, wood, nylon, leaves, etc. from the wastewater. Wastewater from the sand settling tank will be sent to the receiving tunnel and then pumped through a garbage screen (fine filter device), where solids larger than 1mm in size continue to be separated from the wastewater to protect the wastewater treatment plants. equipment in the following water treatment stages. The water then flows to the tank by itself.

At the conditioning tank, the flow and concentration of wastewater will be regulated stably. In the tank, the agitator system will mix well to stabilize the concentration of compounds in the wastewater, the pH value will be adjusted to the optimal parameters for the biological treatment process to work well.

Wastewater is pumped from the conditioning tank into the UASB tank. In the UASB tank, the microorganisms in the anaerobic form will decompose the organic substances in the wastewater (the treatment efficiency of the UASB tank is calculated according to COD, BOD reaches 60 - 80%) into inorganic substances in simple form. and Biogas (CO_2, CH_4, H_2S, NH_3...).

After the UASB tank, the wastewater is led through the anoxic tank and aerotank tank. Anoxic tank combined with aerotank is selected for general treatment: BOD reduction, nitrification, NH_4+ reduction and NO_3- to N_2 reduction, Phosphorus reduction. With the choice of activated sludge tank to combine treatment between anoxic and aerobic treatment processes, it will take advantage of the amount of carbon when reducing BOD, so there is no need to supply additional carbon from the outside when it is necessary to remove NO_3-, saving 50% of oxygen when nitrifying to reduce NH_4+ due to taking advantage of oxygen from NO_3- reduction process.

The concentration of activated sludge in the tank ranges from 1,000-3,000 mg MLSS/L. The higher the activated sludge concentration, the greater the applied organic load of the tank. Oxygen (air) is supplied to the aerotank by highly efficient air blowers and air distribution systems with bubble sizes less than 10µm. The amount of gas supplied to the tank for the purposes of:

(1) Providing oxygen for aerobic microorganisms to convert dissolved organic matter into water and carbon dioxide, organic nitrogen, and ammonia into NO_3- nitrate.

(2) Mixing evenly wastewater and activated sludge create favorable conditions for microorganisms to come into good contact with the substrates to be treated.

(3) Release gases that inhibit the microbial life process. These gases are produced during the process of microbial decomposition of pollutants.

(4) Positively affecting the reproduction process of microorganisms. The organic matter load of the tank during the aerotank treatment phase ranges from 0.32-0.64 kg BOD/m^3.day and night.

Besides converting organic matter into carbonic CO_2 and water H_2O, aerobic bacteria Nitrisomonas and Nitrobacter also oxidize ammonia NH_3 to nitrite NO_2- and finally nitrate NO_3-.

The amount of O_2 oxygen required to completely oxidize ammonia NH_4+ is 4.57g O_2/g N with 3.43g O_2/g used for nitrite process and 1.14g O_2/g NO_2 oxidized.
For each (01)g of ammonia nitrogen (N-NH_3) converted, 3.96g of O_2 is used, and 0.31g of new cells ($C_5H_7O_2N$) are formed, 7.01g of alkali $CaCO_3$ is separated and 0.16g of inorganic carbon is used to form new cells.

The process of denitrification (denitrification) from nitrate NO_3- to gaseous nitrogen N_2 ensures that the nitrogen concentration in the outlet water meets environmental standards. Biological denitrification involves biological oxidation of many organic substrates in wastewater using nitrate or nitrite as electron acceptors instead of oxygen. In the absence of DO or below the DO limit concentration ≤ 2 mg O_2/L (anoxic conditions).

This conversion is carried out by nitrate-reducing bacteria, which account for about 10-80% of the bacterial mass (sludge). The specific denitrification rate ranges from 0.04g to 0.42g N-NO_3^-/g MLVSS.day, the higher the F/M ratio, the greater the denitrification rate.

The water after the cluster of anoxic - aero tanks automatically flows into the settling tank. The sludge is retained at the bottom of the settling tank. Part is recirculated to anoxic tank, part is taken to sludge storage tank. Next, clear water flows through the intermediate tank to prepare for pressure filtration.

The pressure filter tank consists of layers of materials: support gravel, quartz sand and activated carbon to remove dissolved organic compounds, trace elements, substances that are difficult or non-biodegradable and organic halogens. in order to meet the specified requirements.

Seafood wastewater, after passing through the pressure filter tank, will pass through the disinfection tank before the wastewater is discharged into the receiving source.

The sludge in the sludge tank is pumped through the sludge press. Mud cakes are collected and treated by the authorities according to regulations. At the sludge tank, air is supplied to the tank to avoid odors generated by biodegradation of organic matter.

ADVANTAGES OF SEAFOOD PROCESSING WATER WATER TREATMENT TECHNOLOGY:

- The process is capable of high processing efficiency (reaching 98%)
- Capable of treating wastewater with high BOD, reducing nitrogen, phosphorus without adding chemicals.

DISCLAIMER OF SEAFOOD PROCESSING WASTEWATER TREATMENT TECHNOLOGY:

- Operation is complicated, requires a qualified operator.
- Large usable area.

3.3 Apply this project into the current process

This project targets are to reduce the requirements of experiences for the operators, the cost of chemicals as well as the time for used in the process by using sensors and IoT.

First of all, the current process are fixed to specific time and amount of chemicals. These number of time and amount base on experiments in LAB. Meanwhile in real life, these number may vary by the affect of weather and enviroment. Thus, the time might be over/undertaken, and chemicals could be over/under used. The consequences are either the water will be unclarified, or it is a waste of time and chemicals. So, in order to meet the targets, using sensors and IoT are needed.

The first part in the process, which can apply sensors, are when wastewater in the Sand-settling tank. This tank aims are to settle sand and other large, suspended solids. These parameters affect the clarity of water, which is a value can be detected by sensors. Turbidity sensors are used to detemine the clarity of wastewater in this tank. After clarity information are sent to the server, operators can decide when will this water can go into the next part of the treatment process. Treatment-time variable can also be stored for future development of this process.

In the Conditioning tank, it is a must to determine pH level and push it to the neutral level, which is 7. In order to do that, apply chemicals are needed. Using chemicals, which have high pH level like NaOH or low pH level like H_2SO_4, is the way of the current process. But the amount of chemicals is fixed, meaning with X amount of water, apply Y amount of chemicals. This Y is based on experiments taken in the few first process of the factory; this number also can be affected by the types of fishes which are processing. This is a low-effective method. The pH sensor can boost the effectiveness of this process. This sensor measures the water's pH level, send this information to the server for calculating the amount of chemicals needed. However, temperature affects this the measurement, which mean the pH value of a solution is directly dependent on the temperature. Leading to a statement: A pH value without a temperature value is incoherent.

Table 3.3 The value of Kw[1] and pH with increasing temperature

T(^oC)	K_w (mol^2dm^6)	pH
0	0.114×10^{14}	7.47
25	1.008×10^{14}	7.00
50	5.467×10^{14}	6.63
100	51.3×10^{14}	6.14

Table 3.4 Typical pH values for solutions at different temperatures

	0^oC	25^oC	50^oC
Acid	2.01	2.00	2.00
Neutral	7.47	7.00	6.63
Basic	13.80	12.83	12.15

There are two common types of temperature compensation when working with pH measurements.

- ATC compensates for the fluctuating milli-volt output from the electrode. ATC is commonly a built-in today's pH meters for quick and accurate results.
- STC converts the pH at the measurement temperature to the pH at a reference temperature. The reference temperature is generally 25°C. Only pH values taken at the same temperature or converted can be compared to each other.

[1] K_w is water ionisation constant

This will be considered when selecting pH sensor for this project.

The next sensor will be applied is the DO sensor. This sensor detects the amount of dissolved oxygen in water, which will be used to calculate/determine the COD and BOD level. Usually, BOD need a 5-day-measurement, but some of seafood likely to have organic components that degrade very quickly. In such a case, the 5-day BOD and the ultimate BOD would be very similar since there would be very little organic material left after 5 days. Eventually, the needed for a fast calculate/measurement of BOD is existed.

This project will try to use DO sensor to measure the amount of dissolved oxygen in water several times in a fixed period of time. This value will be stored and use for calculating.

In the old method, it is needed to store the samples of wastewater for 5 days for a test. The amount dissolved oxygen exists in the sample after taken minus the amount exist after 5 days have a result, which is the value of BOD_5.

$$DO_{1st} - DO_{5th} = BOD_5(mg/L)$$

But in this project, we will have serveral DO values that are separated by a certain amount of time. The outcome of after measurement results minus previous measurement results will be value of BOD. With enough BOD value, we can reduce error to an acceptable number, and take the mean value as the BOD real value.

$$\frac{(DO_n - DO_{n-1}) + (DO_{n-1} - DO_{n-2}) + \cdots + (DO_2 - DO_1)}{n-1} = BOD\ (mg/L)$$

After that, this value will be used for BOD reducing process in Anoxic tank and with Aerotank.

The last sensor will be applied in this project is TDS sensor. This sensor measure TDS levels in water to indicate the water quality. It will be installed in the pressure filter tank. This sensor will only determine the TDS value for deciding whether if the water meet the specific requirements of dissolved organic compounds, substances that are difficult or non-biodegradable. After this part, the water will flow into next tank for disinfecting process using Chlorines.

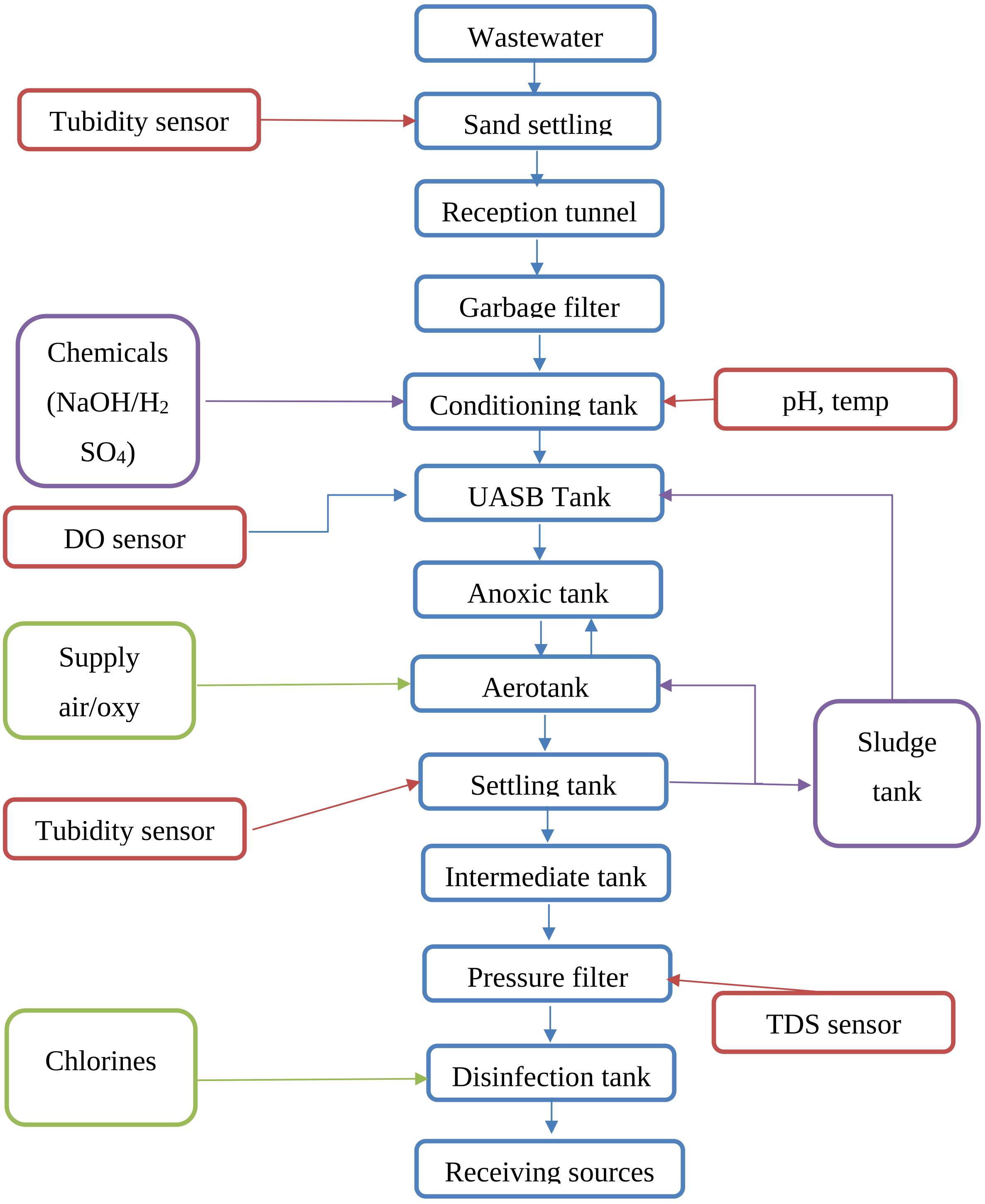


Figure 3.2 Applying this project into current seafood processing wastewater treatment

CHAPTER 4. HARDWARE AND SOFTWARE

This project is aimed to be the type of plug and play devices.

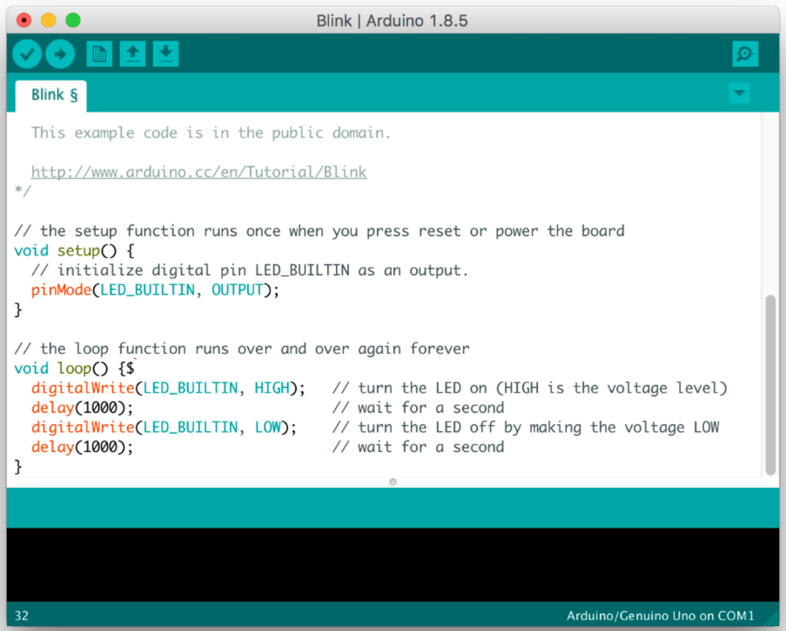
4.1 Software

4.1.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) is a program that is used to write and upload programs to Arduino-compatible boards, as well as other vendor development boards with the support of third-party cores.

ARDUINO

Figure 4.1 Arduino Logo



```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {$
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

```
32                                                    Arduino/Genuino Uno on COM1
```

Figure 4.2 Arduino IDE interface

4.1.2 ThingSpeak

This project will send data to ThingSpeak, an open-source software allows users to communicate with internet enabled devices.

ThingSpeak^{TM} is a cloud based IoT analytics tool that lets you aggregate, visualize, and analyze live data streams. ThingSpeak delivers real-time visualizations of data sent to ThingSpeak by your devices. With the ability to run MATLAB® code in ThingSpeak, you can analyze and handle data as it comes in real time. ThingSpeak is frequently used for IoT system prototype and proof of concept that require analytics.

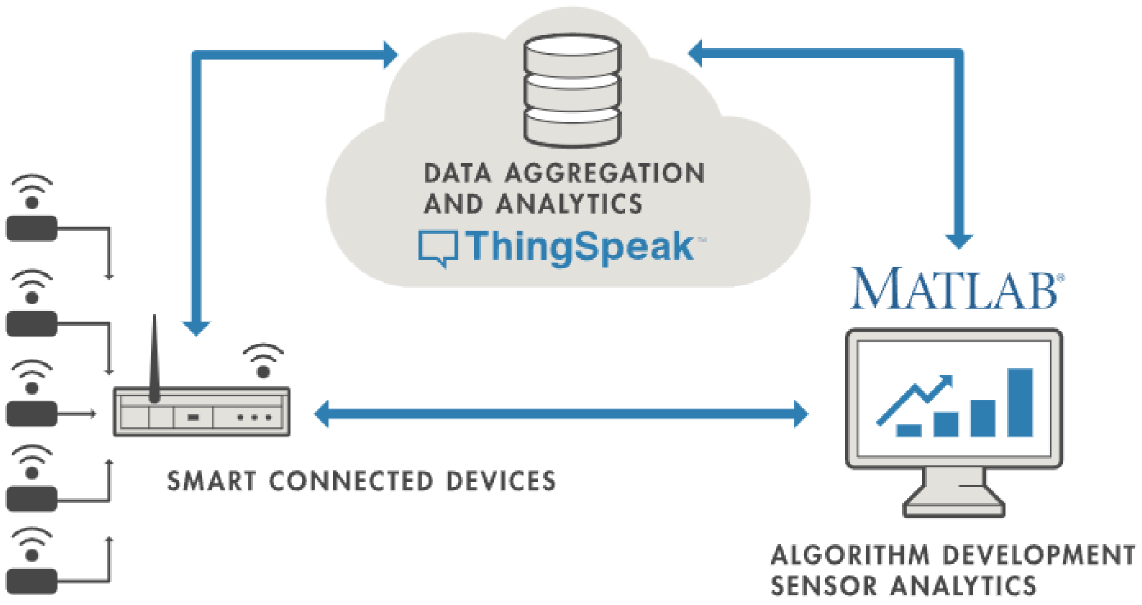


Figure 4.3 ThingSpeak

ThingSpeak Key Features

ThingSpeak allows you to aggregate, visualize and analyze live data streams in the cloud. Some of the key capabilities of ThingSpeak include the ability to:

- Easily configure devices to send data to ThingSpeak using popular IoT protocols.
- Visualize your sensor data in real-time.
- Aggregate data on-demand from third-party sources.
- Use the power of MATLAB to make sense of your IoT data.

- Run your IoT analytics automatically based on schedules or events.
- Prototype and build IoT systems without setting up servers or developing web software.
- Automatically act on your data and communicate using third-party services like Twilio® or Twitter®.

4.2 Hardware

4.2.1 Sensors

4.2.1.1 Dissolved Oxygen

There are several ways to measument dissolved oxygen using sensor.

Optical dissolved oxygen sensors measure the interaction between oxygen and certain luminescent dyes.

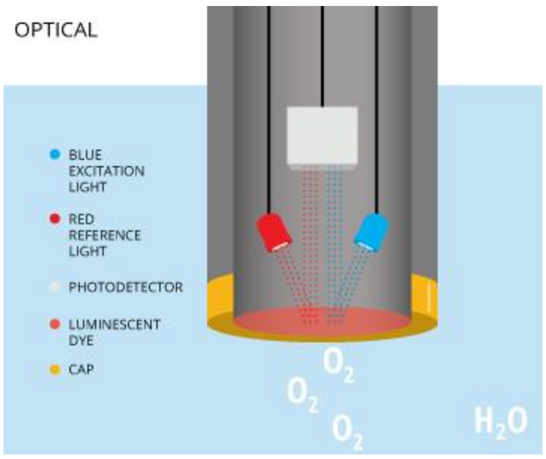


Figure 4.4 Optical DO sensor.

Electrochemical dissolved oxygen sensors can also be called amperometric or Clark-type sensors. There are two types of electrochemical DO sensors: galvanic and polarographic.

- A polarographic DO sensor is an electrochemical sensor that consists of a silver anode and a noble metal (such as gold, platinum or infrequently, silver) cathode in a potassium chloride (KCl) solution. It measurement base on relationship between the oxidation of the silver anode and the reduction of the dissolved oxygen.

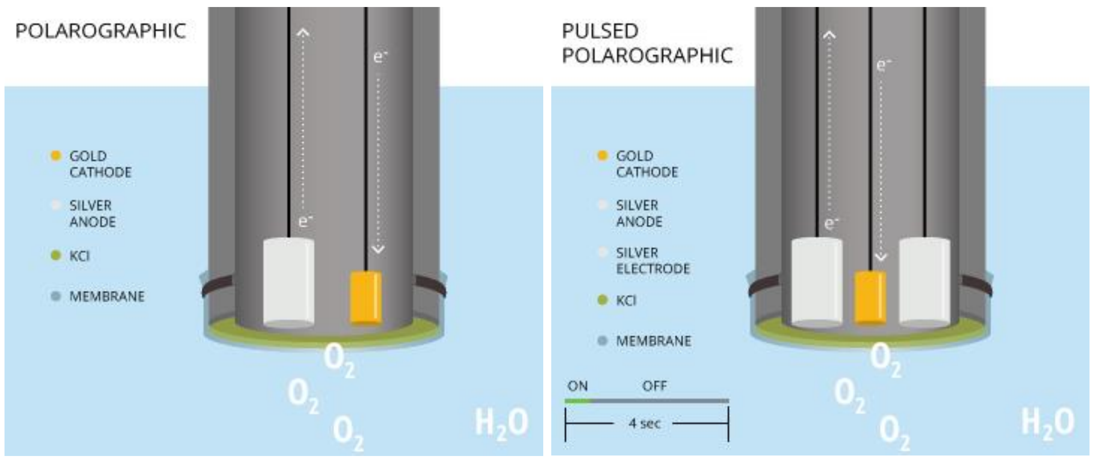


Figure 4.5 Polarographic DO sensors

- Galvanic Dissolved Oxygen Sensors: The anode in a galvanic dissolved oxygen sensor is usually zinc, lead or another active metal while the cathode is silver or another noble metal.

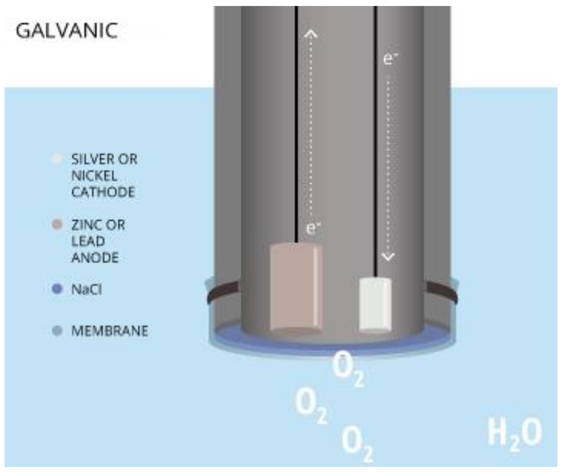


Figure 4.6 Galvantic DO sensor

This project will use a Galvanic type of sensor, which have parameters as below.

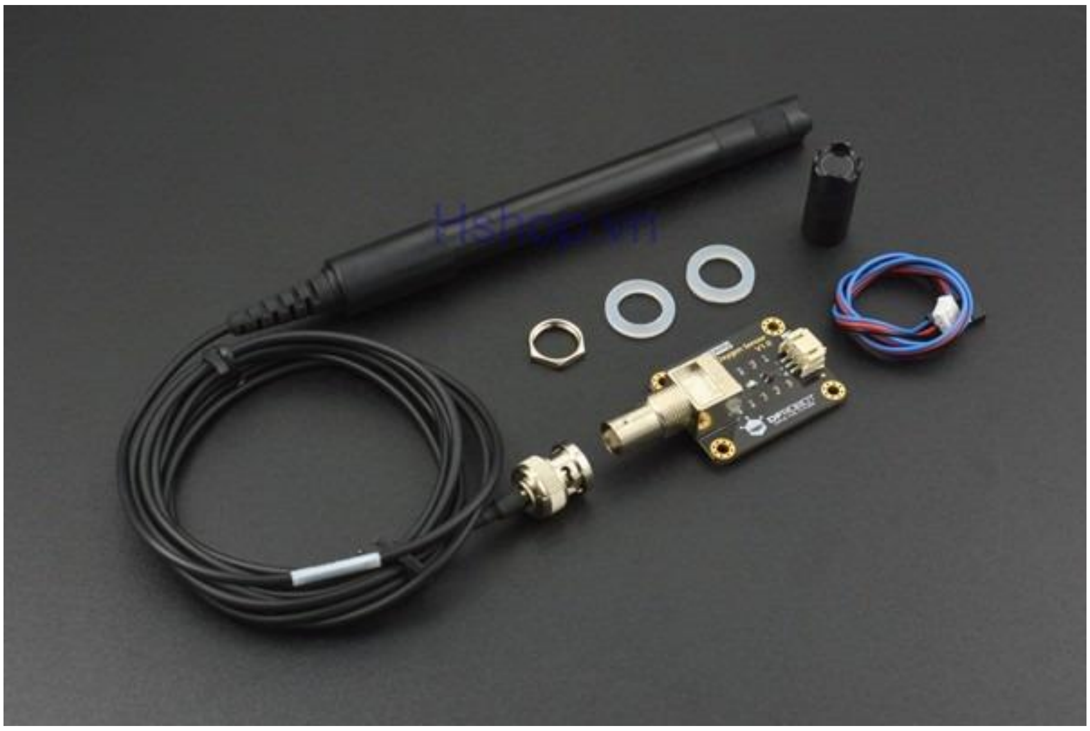


Figure 4.7 Dissolved Oxygen sensor

4.2.1.2 pH Levels sersor

A pH sensor is one of the most important pieces of equipment for determining water quality. This sort of sensor can detect alkalinity and acidity levels in water and other liquids.

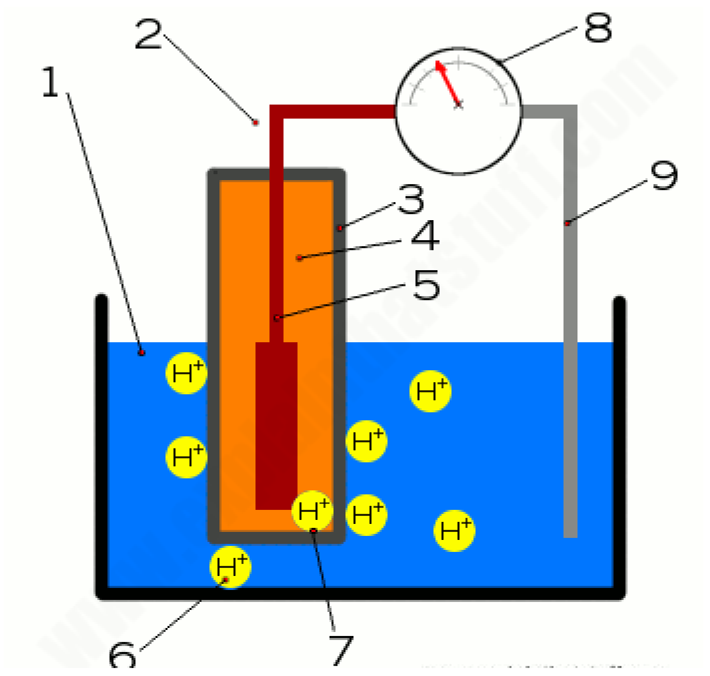


Figure 4.8 How pH sensor work

The liquid (orange) within the glass electrode is a neutral solution with a pH of 7, thus it contains some hydrogen ions (H+). Assume you're evaluating an unknown liquid (blue), which is substantially more acidic and hence includes a higher concentration of hydrogen ions. The glass electrode measures the difference in pH between the orange and blue solutions by comparing the voltages produced by their hydrogen ions. We can calculate the pH of the blue solution using the pH of the orange solution.

Note:

(1) Solution being tested.

(2) Glass electrode.

(3) A thin layer of silica glass containing metal salts.

(4) Potassium chloride solution.

(5) Internal electrode made from silver/silver chloride.

(6) Hydrogen ions formed in the test solution interact with the outer surface of the glass.

(7) Hydrogen ions formed in the potassium chloride solution interact with the inside surface of the glass.

(8) Voltage meter for measuring potential differentce (this value will be converted into a pH reading).

(9) Reference electrode acts as a baseline or reference for the measurement.

Fig ure 4.9 DFRobot's Analog pH sensor

4.2.1.3 Turbidity sensor

One of the preferred methods for turbidity water testing is the use of a nephelometer, or a turbidity meter and probe. A turbidity probe works by sending a light beam into the water to be tested. This light will then be scattered by any suspended particles. A light detector is placed at (usually) a 90-degree angle to the light source, and detects the amount of light that is reflected back at it. The amount of light reflected is used to determine the particle density within the water. The more light that is detected, the more particles are present in the water.

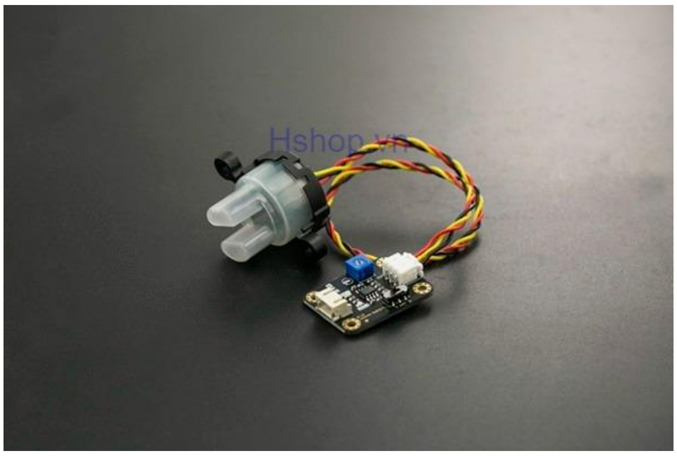


Figure 4.10 Turbiduty sensor

4.2.1.4 Temperature sensor

When a change in temperature is detected, the temperature sensor is made up of two metals that generate electrical voltage or resistance.

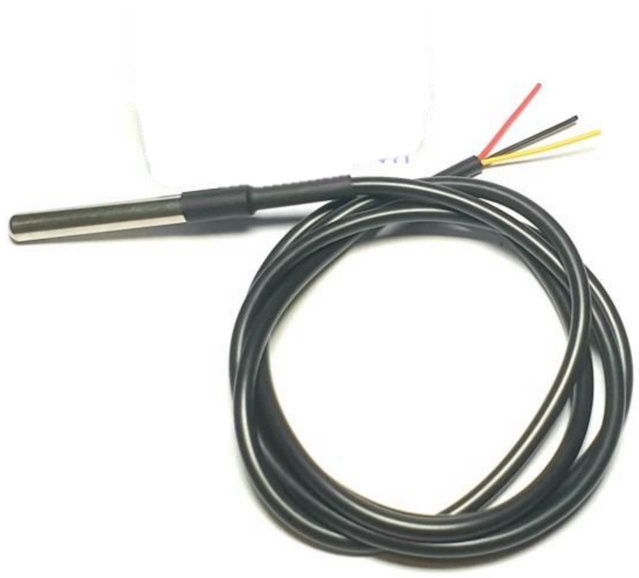


Figure 4.11 Temperature sensor

4.2.1.5 Total Dissolved Solid Sensor

A TDS meter is basically an EC meter whereby two electrodes equally spaced apart are inserted into water and used to measure charge. The result is interpreted by the TDS meter and converted into a ppm figure.

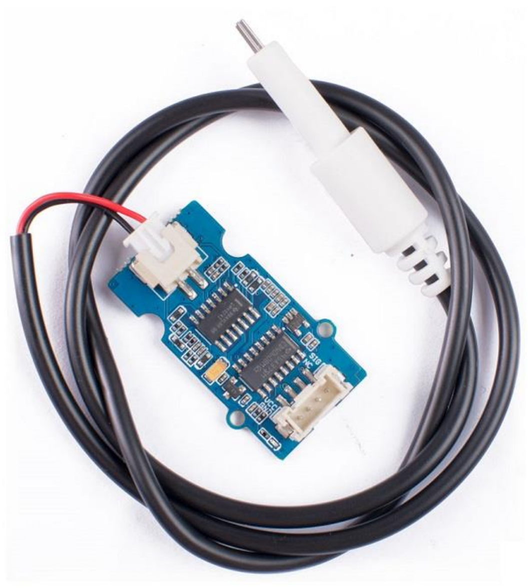


Figure 4.12 TDS sensor

4.2.2 Display

The screen employs I2C connection for line quality and is 0.96 inch OLED screen with I2C interface for gorgeous, opulent, clear visibility during the day and optimal energy savings at an appropriate cost. With only two GPIO pins, the transmission is stable and communication is simple.

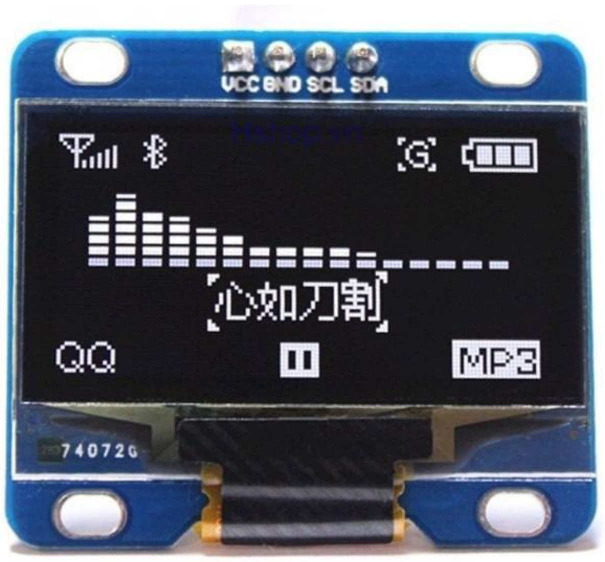


Figure 4.13 0.96 inch OLED screen

4.2.3 Controllers

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs and turn it into an output, such as: knowing when the button is pushed, read something on Twitter and then activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board.

Arduino was created at the Ivrea Interaction Design Institute as a simple tool for rapid prototyping intended for students with no previous experience with electronics or programming. As soon as it gained a larger following, the Arduino board began to evolve in order to meet new needs and problems, evolving from simple 8-bit boards to solutions for IoT, wearables, 3D printing, and embedded settings. All Arduino boards are open source, allowing users to create them on their own and customize them to meet their own needs. The software is also open-source, and it is evolving thanks to contributions from users all over the world.

Arduino has been used in millions of projects and applications due to its simple and accessible user interface. Beginners will find the Arduino software simple to use, but advanced users will find it adaptable. It's compatible with Mac, Windows, and Linux. It is used by teachers and students to create low-cost scientific equipment, to demonstrate chemistry and physics principles, and to begin learning programming and robotics. Designers and architects create interactive prototypes, while musicians and artists utilize it to create installations and try out new instruments. Makers, for example, utilize it to construct many of the items on display at the Maker Faire. Arduino is a valuable tool for learning new skills. Anyone - youngsters, amateurs, artists, programmers - may get started tinkering by following the step-by-step instructions in a kit or sharing ideas with other Arduino members online.

For physical computing, there are a variety of additional microcontrollers and microcontroller platforms to choose from. Similar functionality can be found in Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many other programs. All these programs condense the complicated elements of microcontroller programming into an easy-to-use package. Although Arduino simplifies the process of working with microcontrollers, it has some advantages over other systems for teachers, students, and curious enthusiasts:

- **Inexpensive -** In comparison to other microcontroller systems, Arduino boards are comparatively affordable. The Arduino module's cheapest version can be assembled by hand, and even pre-made Arduino modules cost less than $50.
- **Cross-platform -** The Arduino Software (IDE) is available for Windows, Macintosh OSX, and Linux. The majority of microcontroller systems are only compatible with Windows.
- **Simple, clear programming environment -** The Arduino Software (IDE) is simple to use for beginners, but it's also flexible enough for advanced users. It's built on the Processing programming environment, which is helpful for teachers because students learning to program in that environment will be familiar with how the Arduino IDE works.
- **Open source and extensible software -** The Arduino software is accessible as open-source tools for experienced programmers to extend. People who want to learn more about the language can use C++ libraries, and those who want to learn more about the technical specifics can switch from Arduino to the AVR C programming language. Similarly, if you want to, you can include AVR-C code directly in your Arduino applications.
- **Open source and extensible hardware -** The Arduino board blueprints are made available under a Creative Commons license, allowing experienced circuit designers to create their own version of the module, extending, and upgrading it. Even inexperienced users may put together the breadboard version of the module to learn how it works and save money.

The Seeeduino XIAO is the tiniest of the Seeeduino family members. It is equipped with the ATSAMD21G18A-MU, a low-power microcontroller. This little board, on the other hand, provides good processing performance but uses less power.

In fact, it is designed to be small enough to be employed in wearable gadgets and modest applications. The Seeeduino XIAO includes 14 GPIO PINs that can be used for 11 digital interfaces, 11 fake interfaces, 10 PWM interfaces (D1-D10), 1 DAC output pin D0, 1 SWD pad interface, 1 I2C interface, 1 SPI interface, 1 UART interface, Serial communication indication (T/R), and Blink light (L). Green, yellow, blue, and blue are the colors of LEDs (Power, L, RX, TX). Furthermore, the Seeeduino XIAO has a Type-C interface for power and code download. You can short link the two reset buttons to reset the board.

Table 4.1 Seeeduino's specifications

Item	Value
CPU	ARM Cortex-M0+ CPU(SAMD21G18) running at up to 48MHz
Flash Memory	256KB
SRAM	32KB
Digital I/O Pins	11
Analog I/O Pins	11
I2C interface	1
SPI interface	1
QTouch	7 (A0,A1,A6,A7,A8,A9,A10)
UART interface	1
Power supply and downloading interface	Type-C
Power	3.3V/5V DC
Dimensions	20×17.5×3.5mm

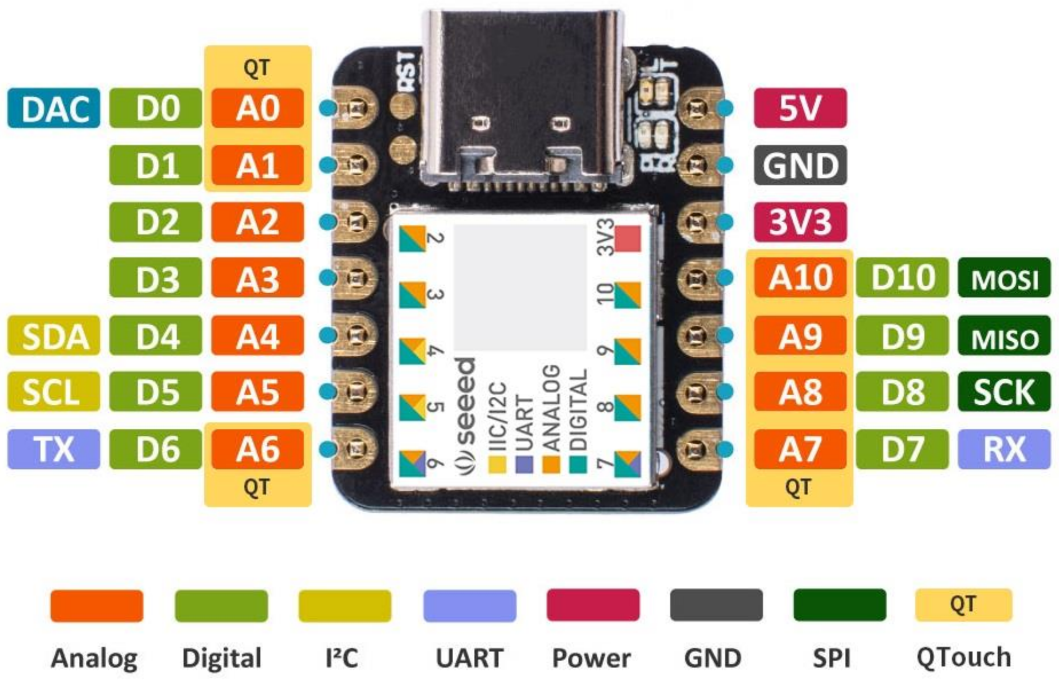


Figure 4.14 Seeeduino's PINs

Pros

- Less space consumption.
- More powerful than other duinos.
- Easy to connect.
- We don't need to configure the pins ourselves, after using the pins, you can call a function directly.

Cons

- Only has 14 pins (excluded power pins then it only has 11).

4.2.4 Module for connection between controllers

Radio frequencies works by transmitting and receiving electromagnetic waves. The radio signal is an electronic current moving back and forth very quickly. A transmitter radiates this field outward via an antenna; a receiver then picks up the field and translates it to the information.

Figure 4.15 NRF24L01 + PA + LNA module

4.2.5 Module for connection between controller and Internet

The ESP8266 UART ESP-01S Ai-Thinker Wifi transceiver circuit is manufactured by Ai-Thinker using the Espressif ESP8266 Wifi SoC IC, used to connect to the microcontroller to perform the function of transmitting and receiving data via Wifi, the circuit has a compact design, uses UART communication with a lot of libraries and sample codes from the community, ESP-01S is used in IoT applications and control devices via Wifi.

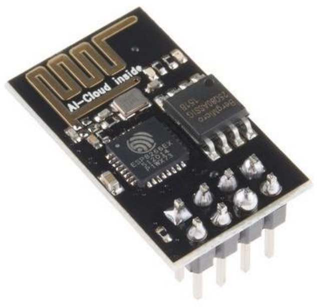


Figure 4.16 ESP-01S

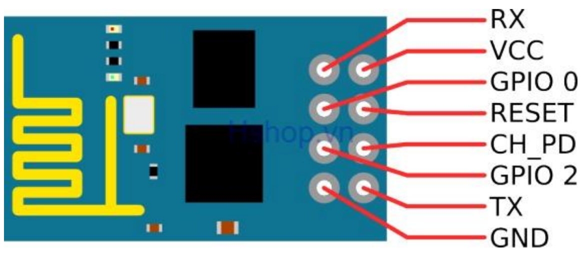


Figure 4.17 ESP-01/ESP-01S pinout

4.2.6 Power Supply

The circuit in this project need 3.3V for operate so we need an adapter for power supply.

Figure 4.18 AC-DC Power Supply 3.3VDC

- Input: AC 100-240V 50/60HZ.
- Output: DC 3.3V - 2A - Standard 5.5mm plug
- 5.5mm plug standard anode in the middle.

CHAPTER 5. MODEL CONSTRUCTION AND RESULTS

5.1 Model construction

This is how data taken and sent to cloud.

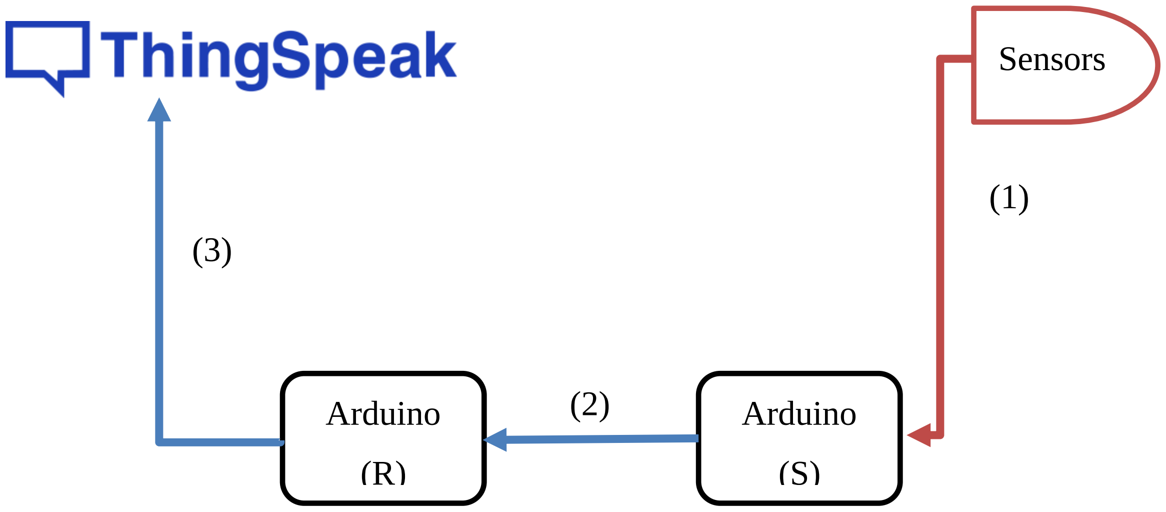


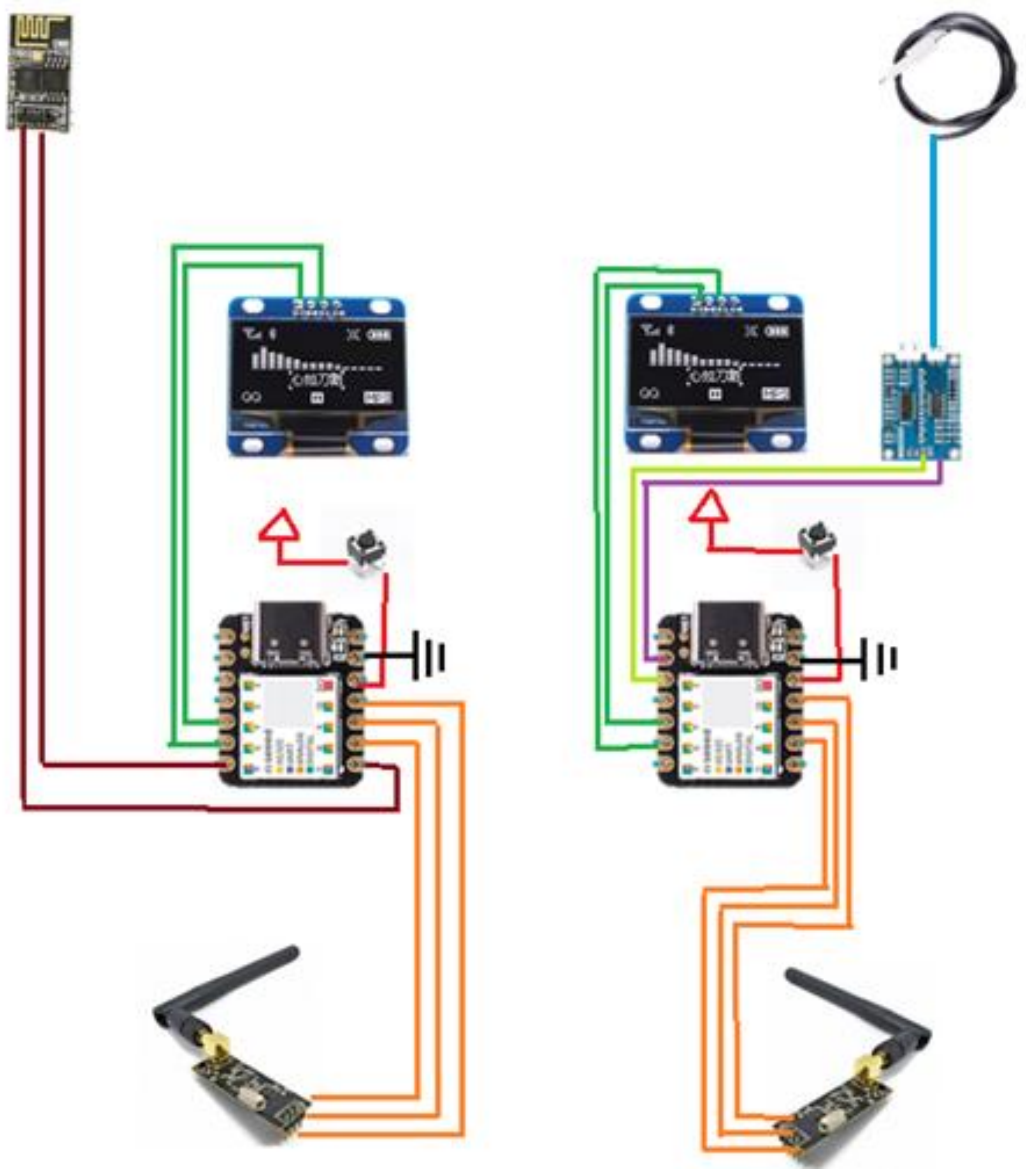
Figure 5.1 Data flow in this project

Note:

(1) Raw data (% voltage level) accquired by sensors, turn it into signal and send to the Sender (Arduino S). Using any types of communication, the sensor designed to (UART/SPI/I2C…).

(2) The Sender reads and converts the information into the desired data (temp…). Thus, send to the Receiver using Radio Frequency through module NRF24L01.

(3) The Receiver take the data and send it to ThingSpeak using ESP8266 through API.



🟥	Power Source (3.3V)	🟧	SPI
⬛	Ground	🟥	UART
🟩	I2C	🟦	Voltage level
🟪	1-Wire	🟩	Voltage level (for Arduino)

Figure 5.2 Wiring for the project

5.2 Results

My result of constructing model is kind of sloppy since it had to tweak a lot and my time I gave is not enough. However, it works out just fine.

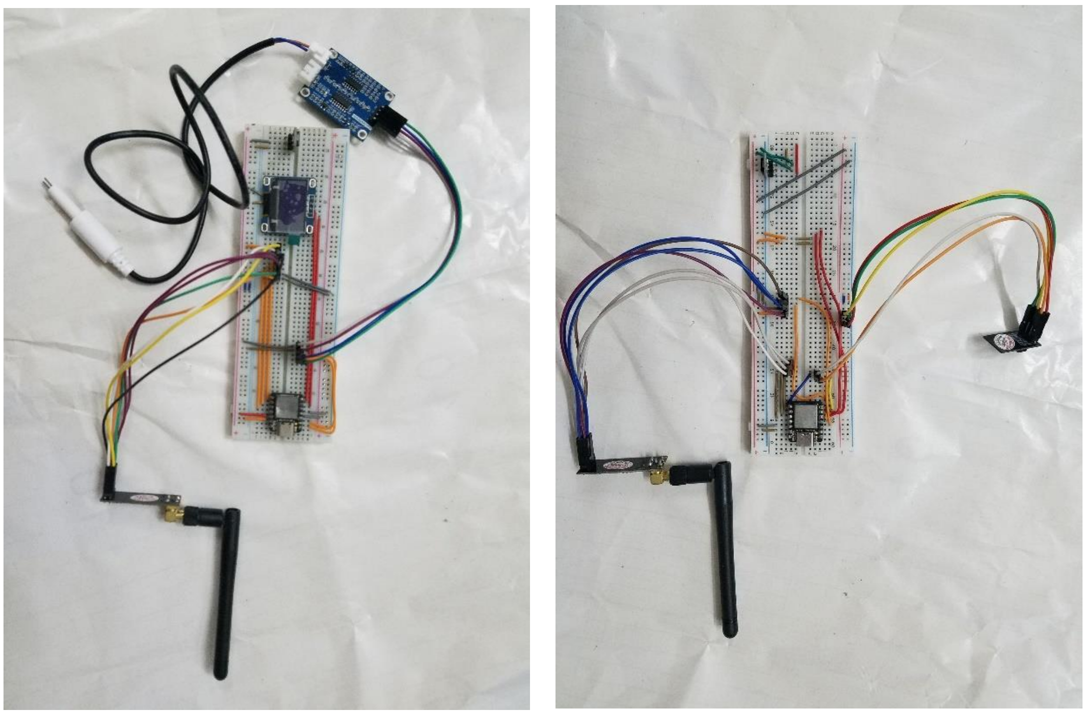


Figure 5.3 Sender (*left)* and Receiver (*right*).

For it hard to take real samples, I randomly create 5 types of samples for testing the model.

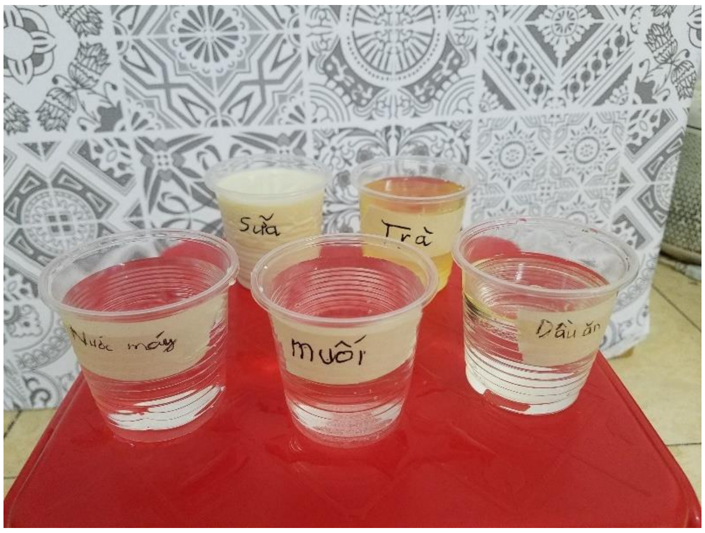


Figure 5.4 Samples (Milk, Tea, Cooking oil, Salt water, Tap water)

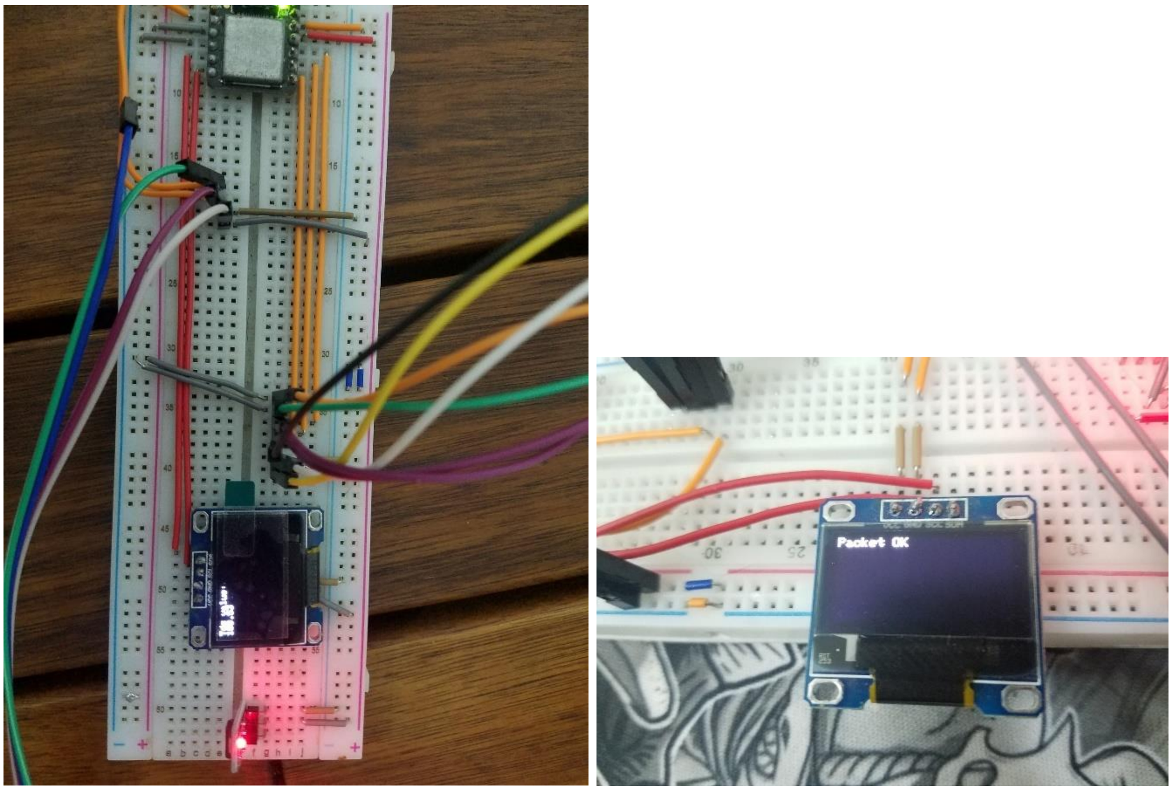


Figure 5.5 OLED screen when Sender (*left*) and Receiver (*right*) are working

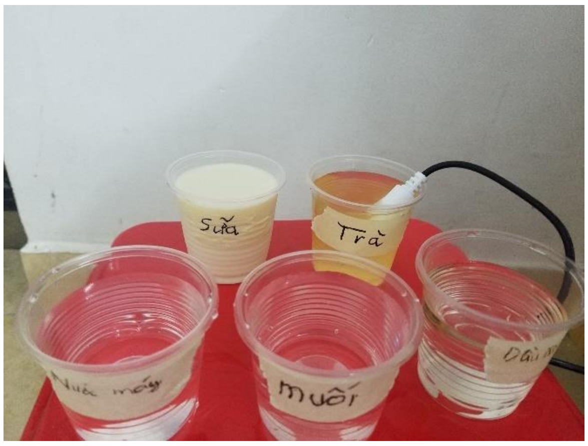
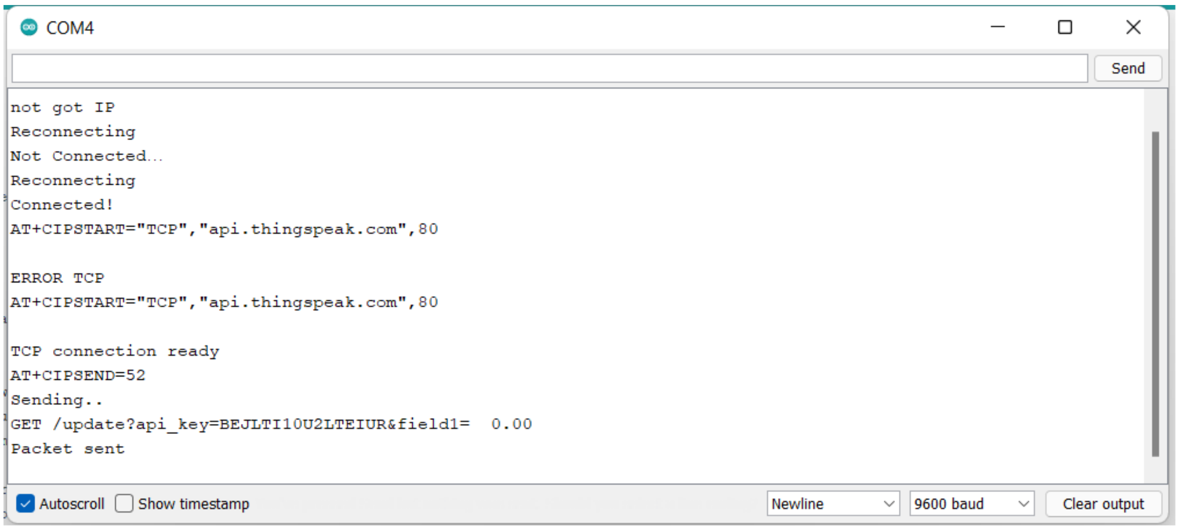


Figure 5.6 Measuring (tea sample)



```
not got IP
Reconnecting
Not Connected...
Reconnecting
Connected!
AT+CIPSTART="TCP","api.thingspeak.com",80

ERROR TCP
AT+CIPSTART="TCP","api.thingspeak.com",80

TCP connection ready
AT+CIPSEND=52
Sending..
GET /update?api_key=BEJLTI10U2LTEIUR&field1=  0.00
Packet sent
```

[✓ Autoscroll] Show timestamp   Newline   9600 baud

Figure 5.7 Serial monitor of Arduino IDE for debugging when working

After all, the results of measurements are sent to Thingspeak. Anyone can view it in my channel[1]. My aim is to take the TDS values of 5 samples in order (milk, tea, cooking oil, salt water and tap water) in total 8 times to see if my model is working right.

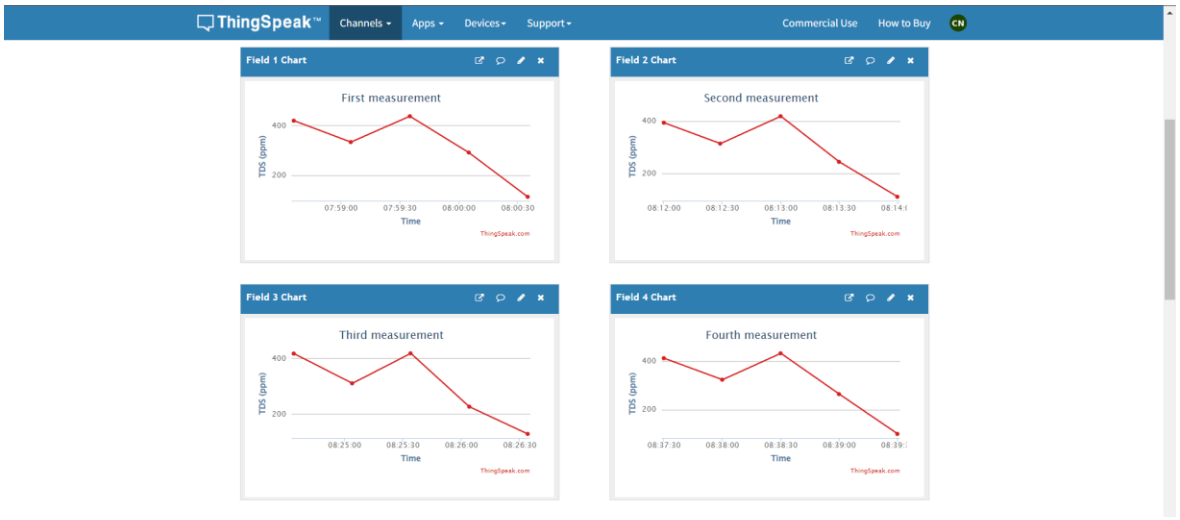


Figure 5.8 My channel to store data (1)

[1] https://thingspeak.com/channels/1560990

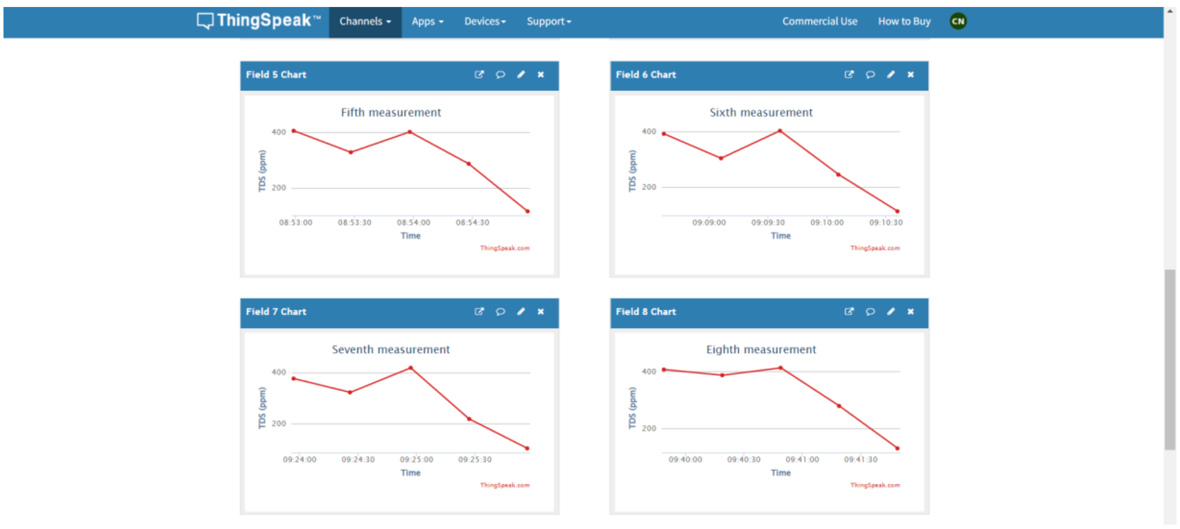


Figure 5.9 My channel to store data (2)

5.2.1 Gains

- The TDS values are obtained.
- Data are able to send and receive from arduino to internet ot to each other.
- Sensor using in this project are common, so they are easy to find.

5.2.2 Obstacles

- Sensors maintance are a big problem since it has to be in the wastewater 24/24.
- Only a few contaminants are able to obtain using sensors.
- Sensors' lifespan also is a problem.

CHAPTER 6. CONCLUSIONS

6.1 Conclusions

The model works, it is able to measure TDS value of samples and send it into the internet. This project might be able to use in real life system.

There are plenty of flaws in this project. The obstacles are hard to deal with since sensors are limited by current technology. Maintainces must be taken frequently.

6.2 Future developments

First of all, this system can only configure the WiFi’s SSID and Password through coding. I want to have some kind of HID module like keyboard, so it going to be able to connect to any WiFi module by the user. And more details will be displayed on the OLED screen.

This system collect data and send it to the internet. Next development needed to be working on is to separate it from Thingspeak. Which mean each wastewater treatment plant should have it own server. Or it can be a server for a network between plants, so others can look at the data and learn from it.

One more development is to be able to control the system through internet. The current is only able to supervise to process, which is not fully IoT.

Lastly, is to use the data which was collected for Machine Learning or Artificial Intelligence. My aim is from the type of seafood, which a specific plant is processing, the system can estimate the entire wastewater treatment proccess without measurement.

REFERENCES

Vietnamese

[1] Thông tư số 77/2015/TT-BTNMT của Bộ Tài nguyên và Môi trường ngày 31/12/2015 về ban hành quy chuẩn kỹ thuật quốc gia về môi trường.

[2] Phuong, M (21/09/2021). *Doanh nghiệp thủy sản phản ứng khi bị xếp vào dạng nguy cơ ô nhiễm cao*. Báo Thanh Niên. Truy cập tại: https://thanhnien.vn/doanh-nghiep-thuy-san-phan-ung-khi-bi-xep-vao-dang-nguy-co-o-nhiem-cao-post1113945.html

English

[3] Martínez, R., Vela, N., el Aatik, A., Murray, E., Roche, P., & Navarro, J. M. (2020). *On the Use of an IoT Integrated System for Water Quality Monitoring and Management in Wastewater Treatment Plants*. Water, 12(4), 1096. doi:10.3390/w12041096

[4] Thomas, O.; Théraulaz, F.; Cerdà, V.; Constant, D.; Quevauviller P. (1997) *Wastewater quality monitoring*. TRAC Trend Anal. Chem. 419–424.

[5] *How Does Temperature Affect pH?*, westlab. Access at https://www.westlab.com/blog/2017/11/15/how-does-temperature-affect-ph

[6] *Learn more – Thingspeak IoT*, Thingspeak. Access at https://thingspeak.com/pages/learn_more

[7] *Seeduino XIAO*, Seed Wiki. Access at https://wiki.seeedstudio.com/Seeeduino-XIAO/

APPENDIX

APPENDIX A. SPECIFICATIONS OF SENSORS/MODULES

This appendix shows specifications of each sensors/modules used in the project. These documentations provided by the manufacturer.

A.1. DO Sensor

- Model: DFRobot Gravity: Analog Dissolved Oxygen Sensor / Meter Kit For Arduino
- Dissolved Oxygen Probe
    - Type: Galvanic Probe
    - Detection Range: 0~20mg/L
    - Response Time: Up to 98% full response, within 90 seconds (25℃)
    - Pressure Range: 0~50PSI
    - Maintenance Period: Membrane Cap Replacement Period: 1~2 months (in muddy water); 4~5 months (in clean water) Filling Solution Replacement Period: Once every month
    - Cable Length: 2 meters
    - Probe Connector: BNC
    - Electrode Service Life: 1 year (normal use)
- Signal Converter Board
    - Operating Voltage: 3.3~5.5V
    - Output Signal: 0~3.0V
    - Cable Connector: BNC
    - Signal Connector: Gravity Analog Interface (PH2.0-3P)
    - Dimension: 42mm * 32mm

A.2. pH Level sensor

- Model: DFRobot Gravity: Analog pH Sensor/Meter Kit V2
- Signal Conversion Board (Transmitter) V2
  - Supply Voltage: 3.3~5.5V
  - Output Voltage: 0~3.0V
  - Probe Connector: BNC
  - Signal Connector: PH2.0-3P
  - Measurement Accuracy: ±0.1@25℃
  - Dimension: 42mm*32mm/1.66*1.26in
- pH Probe
  - Probe Type: Laboratory Grade
  - Detection Range: 0~14
  - Temperature Range: 5~60°C
  - Zero Point: 7±0.5
  - Response Time: <2min
  - Internal Resistance: <250MΩ
  - Probe Life: >0.5 years (depending on the frequency of use)
  - Cable Length: 100cm

A.3. Turbidity sensor

- Operating Voltage: 5V DC
- Operating Current: 40mA (MAX)
- Response Time: <500ms
- Insulation Resistance: 100M (Min)
- Output Method: Analog
- Analog output: 0-4.5V
- Digital Output: High/Low level signal (you can adjust the threshold value by adjusting the potentiometer)
- Operating Temperature: 5℃~90 ℃
- Storage Temperature: -10℃~90℃

- Weight: 30g Adapter
- Dimensions: 38mm*28mm*10mm/1.5inches *1.1inches*0.4inche

A.4. Temperature sensor

- Using voltage: 3~5.5VDC
- Consumption current: 1~1.5mA
- Standard communication: Digital TTL 1-Wire (only 1 Data pin only, note that you need to connect the Data pin of the sensor to a high VCC level through a pull-up resistor of 4k7 Ohm or 10k Ohm before connecting to the microcontroller).
- Measured temperature range: -55~125°C
- Accuracy (error): ±0.5°C
- Resolution: 9~12 bits (settable)
- Response time < 750ms
- Designed with 1m-long waterproof cord with a sturdy protective steel case.

A.5. TDS sensor

- Power supply voltage: 3.3~5VDC
- Consumption current: 3~6mA
- Measured TDS value range: 0~1000ppm, error: ±5%F.S. (25 degrees Celsius)
- Analogue output voltage of TDS: 0~2.3VDC
- Conector TDS Probe: 2-Battery XH-2.54
- Connector Temperature Sensor: 3-Battery XH-2.54
- Dimensions: 42 x 31.2mm

A.6. OLED screen

- Using voltage: 2.2~5.5VDC.
- Power consumption: 0.04w

- Display angle: greater than 160 degrees
- Number of points displayed: 128x64 points.
- Screen width: 0.96 inch
- Display color: White / Blue.
- Communication: I2C
- Driver: SSD1306

A.7. NF24L1+PA+LNA

- Main IC: NRF24L01 + PA + LNA
- Transceiver frequency: 2.4Ghz
- Using voltage: 3.3VDC
- Consumption current: 45mA
- The IO pins are all 5VDC tolerant.
- Communication standard: SPI
- Integrated power amplifier PA (power amplifier) and LNA (Low Noise Amplifier)
- Transceiver power: 20dBm
- Maximum transmission speed: 2Mbit/s
- Standard pin 2 x 4 pin spacing 2.54mm.
- Dimensions: 48 x 15mm

A.8. ESP-01S

- Model: ESP8266 UART ESP-01S Ai-Thinker
- Using voltage: 3.0V~3.6V(Optimal 3.3V)
- Consumption current: Max 320mA (it is recommended to use a separate power supply module for the circuit).
- Supports 802.11 b/g/n standard.
- 2.4 GHz Wi-Fi, supports security standards such as: OPEN, WEP, WPA_PSK, WPA2_PSK, WPA_WPA2_PSK.

- Supports both TCP and UDP communication.
- Standard UART communication with Firmware supports AT Command instruction set, default baudrate 9600 or 115200.
- There are 3 operating modes: Client, Access Point, Both Client and Access Point.
- Dimensions: 24.8 x 14.3mm

APPENDIX B. PINS CONNECTIONS

This appendix show which pins of arduinos connect to which pins of modules/sensors.

Table B.1 Arduino and TDS+Temp sensor pins connections

Arduino Sender	Module TDS + Temp
Pin 1	A0
Pin 2	T1
3.3V	+
GND	-

Table B.2 Arduino and NRF24L1 pins connections

Arduino Sender/Receiver	NRF24L
GND	GND
3.3V	VCC
3.3V	CE
-	CSN
8	SCK
10	MOSI
9	MISO
-	IRQ

*Note: pin CE should set as low when connect to power source and pull high after 2 seconds for protect module lifespan.

Table B.3 Arduino and OLED screen pins connections

Arduino Sender/Receiver	Oled
GND	GND

3.3V	VCC
4	SDA
5	SCL

Table B.4 Arduino and ESP8266 pins connections

Arduino Receiver	ESP8266
GND	GND
3.3V	VCC
7	TX
6	RX
3.3V	RST
3.3V	CH_PD
-	GPIO0
-	GPIO2

APPENDIX C. SOURCE CODE FOR ARDUINO

C.1. Source code for the Sender arduino

```
#include <SPI.h>
#include "RF24.h"
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SoftwareSerial.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET    -1 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for
128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);
```

```
// instantiate an object for the nRF24L01 transceiver
RF24 radio(2, 3); // using pin 7 for the CE pin, and pin 8 for the CSN pin

// Let these addresses be used for the pair
uint8_t address[][6] = {"1Node", "2Node"};

bool radioNumber = 1; // 0 uses address[0] to transmit, 1 uses address[1] to transmit

int tdsPin = A0;
int tdsSensor = 0;
float Voltage = 0;
float tdsValue = 0;



void setup()
  {
    pinMode(A0, INPUT);
    Serial.begin(115200);
    Serial.println("serial begin");

    while (!Serial) {
      // some boards need to wait to ensure access to serial over USB
    }
    if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS))
      {
        Serial.println(F("SSD1306 allocation failed"));
        for(;;); // Don't proceed, loop forever
      }
      display.display();
```

```
   // initialize the transceiver on the SPI bus
  if (!radio.begin())
    {
      Serial.println(F("radio hardware is not responding!!"));
      setupText("radio hardware is not responding!!");
      while (1) {} // hold in infinite loop
    }
    else
    {
      Serial.println(F("Radio hardware is started!!"));
      setupText("radio hardware is started!!");
    }

  radioNumber = 1;
  radio.setPALevel(RF24_PA_LOW);  // RF24_PA_MAX is default.
 // save on transmission time by setting the radio to only transmit the
 // number of bytes we need to transmit a float
  radio.setPayloadSize(sizeof(float)); // float datatype occupies 4 bytes
 // set the TX address of the RX node into the TX pipe
  radio.openWritingPipe(address[radioNumber]);     // always uses pipe 0
 // set the RX address of the TX node into a RX pipe
  radio.openReadingPipe(1, address[!radioNumber]); // using pipe 1
  radio.stopListening();  // put radio in TX mode

} // setup

void loop()
 {
    tdsSensor = analogRead(tdsPin);
    Voltage = tdsSensor*5/1024.0; //Convert analog reading to Voltage
    tdsValue =(133.42/Voltage*Voltage*Voltage - 255.86*Voltage*Voltage +
857.39*Voltage)*0.5; //Convert voltage value to TDS value

    bool reporttds = radio.write(&tdsValue, sizeof(float));     // transmit & save the
report
    if (reporttds)
     {
       Serial.print(F("Transmission successful! "));        // payload was delivered
       setupText("Transmission successful!");
       Serial.print(F("Tds value: "));
       Serial.println(tdsValue);
       setupText("Tds value: ");
       display.println(tdsValue);
       display.display();
     }
    else
     {
       Serial.println(F("Transmission failed or timed out")); // payload was not
delivered
       setupText("Transmission failed or timed out");
       Serial.print(F("Tds value: "));
       Serial.println(tdsValue);
       setupText("Tds value: ");
       display.println(tdsValue);
       display.display();
     }
     // to make this example readable in the serial monitor
    delay(1000);  // slow transmissions down by 1 second
  }

void setupText(String str)
  {
    display.clearDisplay();
    display.setCursor(0, 0);
    display.setTextSize(0.5);
    display.setTextColor(WHITE);
    display.println(str);
    display.display();
  }
```

C.2. Source code for the Receiver arduino

```
#include <SPI.h>
#include "RF24.h"
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SoftwareSerial.h>

// instantiate an object for the nRF24L01 transceiver
RF24 radio(2, 3);
uint8_t address[][6] = {"1Node", "2Node"};

bool radioNumber = 1; // 0 uses address[0] to transmit, 1 uses address[1] to transmit
```

```
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET    -1 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for
128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);



// Variables for senspr and for sending data to Thingspeak
float tdsValue = 0;
unsigned long LastConnection = 10000;
unsigned long CurrentTime = 0;
String tsData;
String value;

char sensorBuffer[10];

// Thingspeak
String ThingspeakAdd = "api.thingspeak.com"; // API
String APIwrite = "****************"; //  APIcode
const int UpdateInterval = 30000; // 30 giây

// Wifi variables
int wifiStatus = 0; // WifiStatus
String ssid = "********"; // Wifi Name
String pass = "********"; // Wifi Password
```

```
void setup() {

    Serial.begin(9600);
    while (!Serial) {
      // some boards need to wait to ensure access to serial over USB
    }
    if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS))
      {
        Serial.println(F("SSD1306 allocation failed"));
        for(;;); // Don't proceed, loop forever
      }

  display.display();
  connectWifi();
  Serial1.begin(115200);
  delay(2000);
  if(Serial1.find("GOT IP"))
    {
      Serial.println("GOT IP");
      setupText();
      display.print("Got IP   ");
      display.display();
      //
      wifiStatus = 1;
    }
  else
    {
      Serial.println("not got IP");
```

```
   }



  // initialize the transceiver on the SPI bus
  if (!radio.begin())
   {
     Serial.println(F("radio hardware is not responding!!"));
     while (1) {} // hold in infinite loop
   }

  radioNumber = 0;

  radio.setPALevel(RF24_PA_MAX);  // RF24_PA_MAX is default.
  radio.setPayloadSize(sizeof(float)); // float datatype occupies 4 bytes

  // set the TX address of the RX node into the TX pipe
  radio.openWritingPipe(address[radioNumber]);     // always uses pipe 0

  // set the RX address of the TX node into a RX pipe
  radio.openReadingPipe(1, address[!radioNumber]); // using pipe 1

radio.startListening(); // put radio in RX mode

} // setup

void loop()
  {
   while (wifiStatus == 0)
     {
       Serial.println("Reconnecting");
       setupText();
       display.print("Reconnecting");
       display.display();
       delay(5000);
       connectWifi();
     }
   // This device is a RX node

   uint8_t pipe;
   if (radio.available(&pipe))
     {             // is there a payload? get the pipe number that recieved it
       uint8_t bytes = radio.getPayloadSize(); // get the size of the payload
       radio.read(&tdsValue, bytes);            // fetch payload from FIFO
       Serial.print(F("Received tds value: "));
       Serial.println(tdsValue);                // print the payload's value
       setupText();
       display.print("TDS value: ");
       display.print(tdsValue);
       display.display();
     }
   // thu thập dữ liệu

  value = dtostrf(tdsValue,6,2,sensorBuffer);

  // đến thời điểm cập nhật
  CurrentTime = millis();
  if(CurrentTime - LastConnection > UpdateInterval)
   {
    //
    LastConnection = millis();
    //
    if(2>1)
      {
        //
        // Data send to Thingspeak
        tsData = "&field1=" + value;
        //
        connectThingSpeak();
        //
        if(Serial1.find("OK") || Serial1.find("ALREADY"))
          {
            Serial.println("TCP connection ready");
            setupText();
            display.print("TCP OK   ");
            display.display();
            //
            updateThingSpeak(tsData);
          }
        else
          {
            setupText();
            display.print("TCP Error ");
            display.display();
            Serial.println("ERROR TCP");
          }
      }
  }
```

```
} // loop

void setupText()
  {
    display.clearDisplay();
    display.setCursor(0, 0);
    display.setTextSize(0.5);
    display.setTextColor(WHITE);
  }

void reset()
  {
    Serial1.print("AT+RST\r\n");
    delay(200);
    //
    if(Serial1.find("OK") )
      {
      Serial.println("Module OK");
      setupText();
      display.print("Serial1 OK   ");
      display.display();
      }
    else
      {
      Serial.println("Module…");
      setupText();
      display.print("Serial1…  ");
      display.display();
      }
  //
  }

void connectWifi()
  {
    String cmd = "AT+CWJAP=\"" + ssid +"\",\"" + pass + "\"";
    cmd += "\r\n";
    Serial1.print(cmd);
    delay(3000);
    //
    if(Serial1.find("OK"))
      {
        Serial.println("Connected!");
        setupText();
        display.print("WIFI OK ");
        display.display();
        //
        wifiStatus = 1;
      }
    else
      {
        wifiStatus = 0;
        //
        Serial.println("Not Connected…");
        setupText();
        display.print("NO WIFI  ");
        display.display();
      }
    //
  }
//
void updateThingSpeak(String tsData)
  {
    //
    String postRequest = "GET /update?api_key=" + APIwrite + tsData + "\r\n";
    String sendCmd = "AT+CIPSEND=";
    Serial1.print(sendCmd);
    Serial1.print(postRequest.length());
    Serial1.print("\r\n");
    //
    Serial.print(sendCmd);
    Serial.println(postRequest.length());
    //
    delay(500);
    //
    if(Serial1.find(">"))
      {
        Serial.println("Sending..");
        setupText();
        display.print("Sending  ");
        display.display();
        //
        Serial1.print(postRequest);
        //Serial1.print("\r\n");
        //
        Serial.print(postRequest);
        delay(1000);
        //
```

```
       if(Serial1.find("SEND OK"))
        {
          Serial.println("Packet sent");
          setupText();
          display.print("Packet OK ");
          display.display();
          //
          Serial1.print("AT+CIPCLOSE\r\n");
        }
       else
        {
          Serial.println("Packet error");
          setupText();
          display.print("Packet…");
          display.display();
          //
          Serial1.print("AT+CIPCLOSE\r\n");
        }
      }
     else
      {
       setupText();
       display.print("NOT SENT ");
       display.display();
       Serial.println("NOT SENT");
      }
 }
//
void connectThingSpeak()
```

```
 {
  String cmdTCP = "AT+CIPSTART=\"TCP\",\"";
  cmdTCP += ThingspeakAdd;
  cmdTCP += "\",80";
  cmdTCP += "\r\n";
  Serial1.print(cmdTCP);
  delay(10);
  Serial.println(cmdTCP);
  delay(100);
 }
```