# A COMPARATIVE STUDY OF ARDUINO UNO R3 AND STM32F411RE SYSTEM PERFORMANCE IN DEVELOPING WATER MONITORING SYSTEM

LIM MENG SHIN

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# A COMPARATIVE STUDY OF ARDUINO UNO AND STM32F411RE SYSTEM PERFORMANCE IN DEVELOPING INTO WATER MONITORING SYSTEM

**LIM MENG SHIN**

**This report is submitted in partial fulfilment of the requirements for the degree of Bachelor of Electronic Engineering with Honours**

**Faculty of Electronics and Computer Engineering**
**Universiti Teknikal Malaysia Melaka**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**2020**

# DECLARATION

I declare that this report entitled "A Comparative study of Arduino UNO and STM32F411RE System Performance in Developing into Water monitoring System" is the result of my own work except for quotes as cited in the references.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

| | | |
|---|---|---|
| Signature | : | |
| Author | : | LIM MENG SHIN |
| Date | : | 27 JUNE 2020 |

# APPROVAL

I hereby declare that I have read this thesis, and in my opinion, this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours.

اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

| | | |
|---|---|---|
| Signature | : | |
| Supervisor Name | : | Dr. Sharatul Izah Binti samsudin |
| Date | : | 27 June 2020 |

# DEDICATION

To my family for supporting me with love and their dedicated partnership.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# ABSTRACT

A microcontroller could be a computer that has been minimized and implement into one metal-oxide-semiconductor (MOS) computer chip. In trending world, various type of microcontroller has been introduced with progressively advanced CPUs (Central Processing Units) well as associate degree integrated knowledge memory (usually SRAM), a program memory (usually FLASH) and a spread of peripherals. In this project, several experiments have been done through Arduino and STM 32 microcontrollers to compare and study these two different platform microcontrollers' system performances. To keep up the trend of the Internet of Things (IoT), in this project, the Arduino and STM32 board are developed and tested to a water monitoring system which equipped with data transferred capabilities over a network without requiring human-to-human or human-to-computer interaction. The result of study references to students as well as to the public and industries related to significant advances in microcontroller development.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# ABSTRAK

Mikropengawal boleh menjadi komputer yang telah meminimumkan dan dilaksanakan menjadi satu cip komputer logam-oksida-semikonduktor (MOS). Di dunia yang berkembang, pelbagai jenis mikropengawal telah diperkenalkan dengan CPU maju (Unit Pemprosesan Pusat) dan juga ijazah bersekutu memori pengetahuan bersepadu (biasanya SRAM), memori program (biasanya FLASH) dan penyebaran peranti. Dalam projek ini, beberapa eksperimen telah dijalankan melalui mikropengawal Arduino dan STM 32 untuk membandingkan dan mengkaji kedua-dua platform sistem mikropengawal yang berbeza. Untuk mengekalkan trend Internet of Things (IoT), dalam projek ini, papan Arduino dan STM32 telah dibangunkan dan diuji kepada system pemantauan air yang dilengkapi dengan keupayaan pemindaan data atas talian tanpa memerlukan interaksi manusia ke manusia dan manusia kepada computer. Hsial kajian ini berfungsi sebagai rujukan kepada pelajar serta kepada orang awam dan industry yang berkaitan dengan kemajuan dalam pembangunan mikropengawal.

# ACKNOWLEDGEMENTS

The work presented in this report could not have been completed without the help of numerous people. First and foremost, I would like to convey my gratefulness to my final year project supervisor, Dr. Sharatul Izah Bte Samsudin for giving me his guidance, motivation and invaluable discussion throughout the project. Despite she is being extraordinarily busy with his jobs and duties, she is still managed to guide me along.

Furthermore, I would also have to appreciate the guidance given by other supervisor as well as the panels who share their expertise and recommendation during the project presentation and give me a right direction toward completion of my project.

Besides, I wish to dedicate my deepest thanks and appreciation to my family, girlfriend, and course-mates for their cooperation, encouragement, constructive suggestion and full of support not only for the report completion but also for my entire study in Universiti Teknikal Malaysia Melaka.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

And finally, my sense of gratitude express to Universiti Teknikal Malaysia Melaka and also my PA, Dr. Hazli Rafis Bin Abdul Rahim, for great commitment and cooperation during my Final Year Project.

اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# TABLE OF CONTENTS

UTeM
اونيۏرسيتي تيکنيکل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# LIST OF FIGURES

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UTeM
اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# LIST OF TABLES

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# LIST OF SYMBOLS AND ABBREVIATIONS

| | | |
|---|---|---|
| **Wi-Fi** | **:** | **Wireless Fidelity** |
| **RISC** | **:** | **Reduced Instruction Set Computer** |
| **CISC** | **:** | **Complex Instruction Set Computer** |
| **pH** | **:** | **Potential of Hydrogen** |
| **IoT** | **:** | **Internet of Things** |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# CHAPTER 1

# INTRODUCTION

This chapter described the background of the project. Besides, the problem statement and objectives of this project briefly explained in this chapter. The scope and structure of the report also included in this section.

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 1.1 Project Background

Performance testing is a process to determine the rate of speed, responsiveness, and reliability of a system, a network, a software program, or a workload unit.[1] Microcontroller system performance is mostly affected by the complexity of architectures (RISC & CISC), word size of the machine, the CPU speed, flash memory[2][3]. Over the last decade, there is rapid growth in technologies which approach to development and advancement of the system performance of microcontroller. Past research focuses on classifying the number of bits, memory devices, instruction sets, memory architecture[4]. Besides from organising method, from the previous research, they also focused on developing an optical tomography controller unit. They implemented the industry 4.0 feature which developed an IoT controller unit to compare and study the system performance of two different microcontroller boards.[5][6]

The project aims to study and compare the performance Arduino UNO R3 and STM32F411RE microcontrollers. This experiment is purposely to test the speed and responsiveness of both microcontrollers' system. All the assigned parameters reading will be recorded and analyzed to determine the most efficient and effective microcontroller among Arduino and STM32. To further improve, this project will also focus on the development of Arduino and STM32 microcontrollers board into a water monitoring device with IoT features.

## 1.2 Problem Statement

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

With the vigorous and rapid development of the technologies, various types of microcontrollers introduced with specific system performance. Therefore, to pick the best hardware, it is crucial to acknowledge concepts relating to microcontrollers.[7] The Arduino and STM32 microcontrollers have become a discussion all over the world, especially engineers. Although they are almost similar or having some identical specifications, still in some form factor, there will be slightly different, which will affect the microcontroller's system performance. Many studies had been carried out, such as microcontroller platforms, programming language, instruction set, memory architecture, and types of microcontrollers [4]. Since, the microcontroller is a very powerful integrated circuit designed with memory, processor, and input/output (I/O) peripherals on a single chip to carry out a specific operation in an embedded system that abundantly identified with measurable data. By carrying out the reviews on the system performance in four main performance test experiments, a better comprehension comparative study of Arduino and STM32 development board in a water quality monitoring system done

Besides, the microcontroller also gives a significant influence on our daily life from the shadow from all aspects (washing machine, digital watch, Bluetooth speakers, microwave, remote control)[3][8]. Especially at a tremendous rate of population growth, the human community started to face the anger of water shortages. They elevated by uncontrolled urbanization and industrialization, which pollute the meagre quantity available for use. Today freshwater resource quality and availability are the most pressing environmental challenges in the country[9]. Other than that factor, on the market, there are many costly pool automation systems. A typical pool system will cost up to R 25,000, and most of these systems only operate with saltwater pools with chlorinators. The automation of swimming pools that control acid and chlorine pools is even more costly[10]. Thus, the comparative study of Arduino and STM32 development board in a water quality monitoring system is beneficial. Due to the comparative study are more to compare the system performance of Arduino and STM32 microcontrollers board. The result obtained is capable of providing useful information to the community, especially to engineer when choosing their right and suitable microcontroller that could meet their project requirement. Therefore, a comparative study of Arduino and STM32 development board in water quality monitoring system which is able to provide the overall system performance of Arduino and STM32 will be the useful for the public citizens in selecting a proper and precise microcontroller.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 1.3 The objective of the Research

The purpose of this comparative study is to compare, study, design, and develop the water quality monitoring system. The following objectives of the project determined as:

i. To design the wireless connection stability and latency Arduino UNO R3 and STM32F411RE in the IoT water monitoring system.

ii. To analyze and compare the Arduino UNO R3 and STM32F411RE microcontrollers' speed responsiveness with RISC and CISC instruction set.

## 1.4 Scopes of Work

UTeM
اونيۏرسيتي تيکنيکل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

In this project, two different platforms of microcontrollers which are Arduino UNO R3 and STM32F411RE microcontrollers used to compare and study the system performance of each microcontroller in developing smart water monitoring system. Before that, both microcontrollers will be tested the system performance without producing it into a water monitoring system. The purpose of doing so is to study the actual capability of the system performance of the microcontroller itself by isolating any other external peripheral that will affect towards system performance of the microcontroller. Hence, the latency test and integer arithmetic process introduced. After that, only both microcontrollers developed into the water monitoring system which implemented the IoT feature. The aims are to test both microcontrollers by applying them into a real-life application and determine the stability and latency of the wireless performance of microcontrollers. Two parameters which are pH and temperature, will be measured. The measured values will then be analyzed according to time to determine the graph versus time. From there, the time response analysis applied to determine the system performance of microcontrollers. The software that used to program the Arduino UNO R3 and STM32F411RE is Arduino IDE and Mbed compiler respectively. For time response analysis, the software that will apply in Excel. Finally, both microcontrollers can support or operate up to 32-bit.

## 1.5 Report Structure

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

The thesis organized and separated into five major chapters. In chapter 1, the overview of the comparative study of Arduino and STM32 development discussed in the project background. Besides, the problem statement, objective, and scope of the project will be outlined clearly in this section. In chapter 2, the past studies related to the comparison of different microcontrollers are reviewed and criticized. The background theory of the microcontroller and water monitoring system included in this chapter. Detail discussed in chapter 3, all relevant experiments and techniques used in the project. The flowchart for the smart water monitoring system design and system performance experiment explained in this chapter. In chapter 4, the result of the project will be recorded and interpreted in this section. The obtained and collected data will be analyzed carefully to verify whether the objectives achieved in the last chapter, a conclusion drawn from the project—besides, the recommendation for the plan related to the project made in this section.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# CHAPTER 2

# BACKGROUND STUDY

UTeM
اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

This chapter will discuss the background studies or literature review on topics that are related. The discussions in this chapter are supported by the knowledge that refers to books, journals, articles, and papers.

## 2.1 Microcontroller

Nowadays, the development of technologies leads to the electronic gadgets which getting to be smaller, adaptable, and modest that can do more capacity when contrasted with their forerunners that happened to cover more space, turned out exorbitant with the ability to perform fewer capabilities. The multiple manufacturers of hardware and software that take place in the market today divided into two major categories. It includes the development of products focusing on hobbyists, i.e. equipment and libraries for the Integrated Development Environment (IDEs) developed for Arduino or Arduino-like. The latter products compile code (mostly C language) and hardware, designed to help engineers speed up the process of development[11].

### 2.1.1 Arduino UNO R3

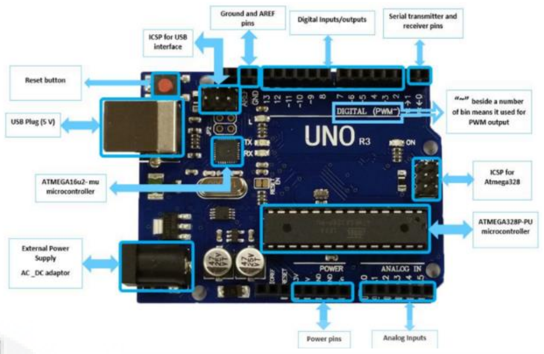

Ground and AREF pins
Digital inputs/outputs
Serial transmitter and receiver pins
ICSP for USB interface
Reset button
USB Plug (5 V)
"~" beside a number of bin means it used for PWM output
ATMEGA16u2- mu microcontroller
ICSP for Atmega328
ATMEGA328P-PU microcontroller
External Power Supply AC _DC adaptor
Power pins
Analog Inputs
UNO R3
DIGITAL (PWM~)
POWER
ANALOG IN

**Figure 2.1 Architecture structure of Arduino Uno.**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

The Arduino Uno is a microcontroller board dependent on the ATmega328 [12]. It has 14 computerized input/yield pins (of which six utilized as PWM yields), six simple sources of info, a 16 MHz clay resonator, a USB association, a power jack, an ICSP header, and a reset catch. It contains everything expected to help the microcontroller; basically, associate it to a PC with a USB link or power it with an AC-to-DC connector or battery to begin. The Uno varies from every former board in that it doesn't utilize the FTDI USB-to-sequential driver chip. Rather, it includes the Atmega16U2 (Atmega8U2 up to rendition R2) customized as a USB-to-sequential converter.

#### 2.1.1.1 Power

The Arduino Uno can be driven with an external power supply or via a USB connection. External (non-USB) control can come from either a wall-wart (AC-to-

DC) or a battery adapter. The connector can be attached by plugging into the power jack of the panel a 2.1 mm centre-positive plug. Battery leads can be placed in the Control connector's GND and Vin pin headers.

The panel will work on a 6 to 20-volt external supply. Nevertheless, when filled with less than 7V, the 5V pin would provide fewer than 5 volts, and the panel could be volatile. The voltage regulator can overheat and damage the panel when using more than 12V. The suggested range is between 7 and 12 volts. The panel will work on a 6 to 20-volt external supply. Nevertheless, when filled with less than 7V, the 5V pin would provide fewer than 5 volts, and the panel could be volatile. The voltage regulator can overheat and damage the panel when using more than 12V. The suggested range is between 7 and 12 volts.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

The power pins are:

- VIN. When using an external power source, the input voltage to the Arduino board (as opposed to 5 volts from the USB connection or other controlled power source). You may supply voltage through this button, and reach it through this pin while supplying voltage through the control socket.
- 5V. This pin provides the board with a controlled 5V from the regulator. The panel can be driven either from the board's DC power jack (7-12V), USB connector (5V), or the board's VIN pin (7-12V). The flow of voltage through the 5V and 3.3V pins bypasses the regulator and can destroy the panel. We're not suggesting this.
- 3V3. The on-board regulator produces a 3.3-volt demand. The maximum current draw is 50 mA.
- GND. Ground pins.

#### 2.1.1.2 Memory

The ATmega328 has 32 KB (with the bootloader utilizing 0.5 Mb). It also has SRAM of 2 KB and EEPROM of 1 KB

#### 2.1.1.3 Input and Output

With pinMode (), digitalWrite(), and digitalRead(), each of the 14 digital pins on the Uno can be used as an input or output. They're running at 5 volts. Can will supply or receive up to 40 mA and has a 20-50 k Ohms internal pull-up resistor (default disconnected). However, many pins have special features:

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

- Serial: from 0 (RX) to 1 (TX). Used to collect (RX) or serial information from (TX) TTL. Such pins are wired to the ATmega8U2 USB-to-TTL Serial chip's corresponding pins.
- Internal breakdown: 2 and 3. Such pins can be set to trigger a low-value interrupt, a rising or falling rim, or a value switch.
- PWMs: 3, 5, 6, 9, 10, 11. Provide analogWrite() feature for 8-bit PWM performance.
- SPI: 10 (SS), 11 (MOSI), 13 (SCK). Using the SPI catalogue, these pins enable SPI interaction.
- LED: 13. It is a built-in LED that connected to digital pin 13. When the input pin is HIGH, the LED is turn on, when the pin is LOW, it's turned off.

The Uno has six analogue inputs, numbered A0 through A5, each with a resolution of 10 bits (i.e. 1024 different values). By definition, they scale from the ground to 5 volts, although the AREF pin and analogReference() feature which can be used to adjust the upper end of their distance. However, many pins have special features:

- TWI: pin A4 or SDA and pin A5 or SCL. Join the Wire collection of TWI contact.

  There are a few other pins on the board:

- AREF: analogue input reference voltage. Used with analogReference().
- Reset: Bring this line LOW to reset the microcontroller. Usually used to add shields with a reset button that blocks the one on the board.

### 2.1.2 STM32F411RE

The STM32F411xC/xE features high-speed integrated storage (up to 512 KB of Flash memory, 128 KB of SRAM), and a wide range of upgraded I /Os and peripherals attached to two APB buses, two AHB buses, and a 32-bit multi-AHB bus matrix.[13]

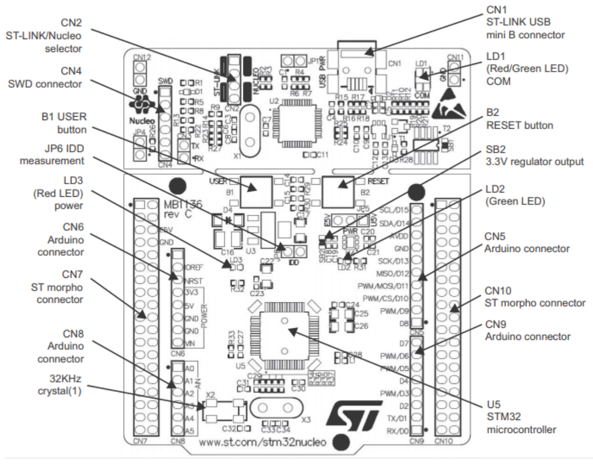


**Figure 2.2 Architecture structure of ST32F411RE.**

Both systems are fitted with one 12-bit ADC, one low-power RTC, six 16-bit general-purpose timers plus one motor control PWM timer, two 32-bit general-purpose timers. These also have default and sophisticated frameworks for contact.

- Up to three I2Cs
- Five SPIs
- Five I2Ss out of which two are full-duplex. To achieve audio class accuracy, the I2S peripherals can be clocked via a dedicated internal audio PLL or via an external clock to allow synchronization.
- Three USARTs
- SDIO interface
- USB 2.0 OTG full speed interface

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 2.1.2.1 Features

- Dynamic Efficiency Line with BAM (Batch Acquisition Mode)
  - 1.7 V to 3.6 V power supply
  - - 40°C to 85/105/125 °C temperature range
- Core: Arm® 32-bit Cortex®-M4 CPU with FPU, Adaptive real-time accelerator (ART Accelerator™) allowing 0-wait state execution from Flash memory, frequency up to 100 MHz, memory protection unit, 125 DMIPS/1.25 DMIPS/MHz (Dhrystone 2.1), and DSP instructions.
- Memories
  - Up to 512 Kbytes of Flash memory

- 128 Kbytes of SRAM

- Clock, reset and supply management
  - 1.7 V to 3.6 V application supply and I/Os
  - POR, PDR, PVD, and BOR
  - 4-to-26 MHz crystal oscillator
  - Internal 16 MHz factory-trimmed RC
  - 32 kHz oscillator for RTC with calibration
  - Internal 32 kHz RC with calibration

- Power consumption
  - Run: 100 µA/MHz (peripheral off)
  - Stop (Flash in Stop mode, fast wakeup time): 42 µA Typ @ 25C; 65 µA max @25 °C
  - Stop (Flash in Deep power-down mode, slow wake-up time): down to 9 µA @ 25 °C; 28 µA max @25 °C
  - Standby: 1.8 µA @25 °C / 1.7 V without RTC; 11 µA @85 °C @1.7 V
  - V_BAT supply for RTC: 1 µA @25 °C

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 2.1.2.2 Batch Acquisition Mode

Throughout software batching, the batch processing mode allows increased power performance. This enables data acquisition directly to storage using the DMA through any interaction peripherals in reduced power usage as well as data processing while the rest of the system is in low power mode (including flash or ART). In an audio system, for instance, a clever mix of PDM audio sample amplification and storage directly from the I2S to RAM (memory or ARTTM stopped) with the DMA utilizing BAM accompanied by some very fast flash processing enables the application's power consumption to be drastically reduced.[13]

#### 2.1.2.3 Embedded Flash Memory

Required for storing programs and files, the devices insert up to 512 Kbytes of Flash memory. The Flash memory can also be turned off in Run and Sleep mode to reduce power consumption. There are two modes available: Flash in Stop mode or Deep Sleep mode (to switch off between power savings and start time).[13]

##### *(a)* One-time Programmable bytes

There is a one-time programmable area with 16 32-byte OTP blocks.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 2.1.2.4 CRC (cyclic redundancy check) calculation unit.

The computation system of the CRC (cyclic redundancy check) is used to acquire a CRC code from a 32-bit information term and a fixed polynomial generator. CRC-based techniques are used, among other applications, to verify data transmission or storage integrity. These provide a way for checking the reliability of the Flash memory in the context of the EN / IEC 60335-1 specification. The CRC calculation system helps measure a code signature during runtime, compared to a link-time produced reference signature and stored at a specified memory position.[13]

#### 2.1.2.5 Clock and Startup

The central CPU clock is chosen when the 16 MHz internal RC oscillator is reset. The 16 MHz internal RC oscillator is factory-controlled to provide a precision of 1 percent at 25 ° C. Then the program may pick either the RC oscillator or an internal clock origin of 4-26 MHz as the device clock. For malfunction, this clock can be tracked. The system automatically switches back to the internal RC oscillator when a failure is identified and a program interrupt (if enabled) is created. The output of the clock is fed into a PLL allowing the frequency to be increased up to 100 MHz. Similarly, full PLL clock entry interrupts management is available when necessary (e.g. if an external oscillator used indirectly fails).

The specification of the two AHB buses, the high-speed APB (APB2), and the low-speed APB (APB1) domains are achievable through several pre-scalers. The two AHB buses ' maximum frequency is 100 MHz while the high-speed APB domains ' maximum frequency is 100 MHz. The total allowed low-speed APB domain frequency is 50 MHz. The tools have a dedicated PLL (PLLI2S) which enables the quality of the audio group to be accomplished.[13]

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.1.3 Comparison of Microcontroller

Recently, there are several competitive microcontroller boards have been introduced and offering something unique and some clone brands. ST Microcontroller is one of the newly introduced microcontrollers that is very popular nowadays.

NUCLEO development boards where the processing is much faster and more memory provided.[14]

The features about NUCLEO boards:

- Mbed compatible.
- Compatible to Arduino as the pins configure provided full access to all Arduino shields.
- The ability to remove the board's programming and debugging area

There are three versions of ST NUCLEO boards, which are Nucleo-32, Nucleo-64, and Nucleo-144. Each of these boards provides different system performance as they consist of varying flash size (bytes) memory.

UTeM
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

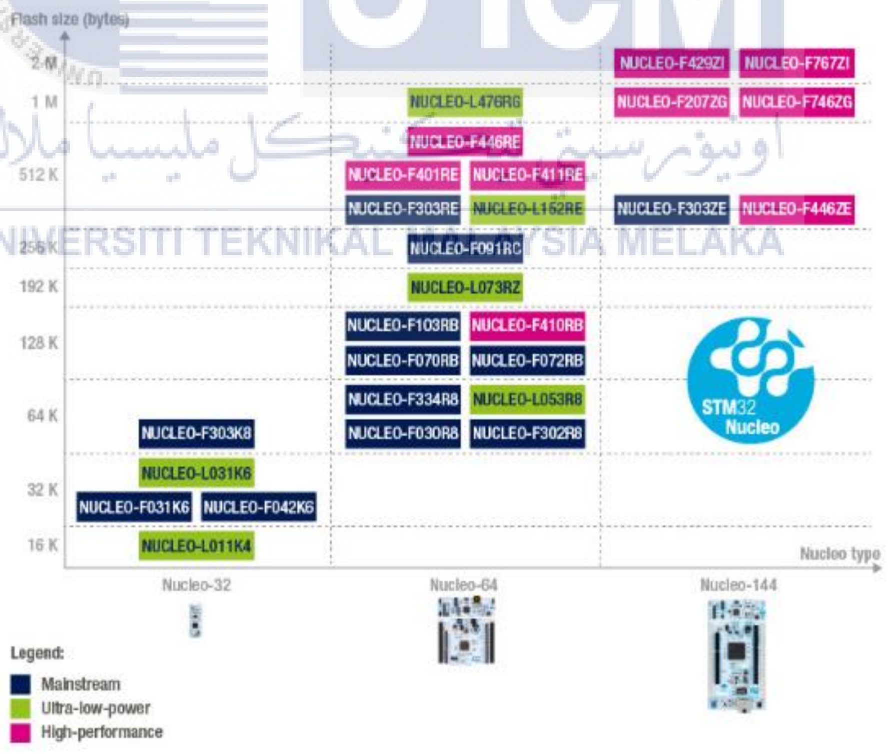


**Figure 2.3 STM NUCLEO Range.[14]**

Although there are three different versions of the NUCLEO board, there are based on ARM Cortex-M, adopting a 32-bit RISC architecture. On the board, two double strips of male connect externally used in other STM dev boards known as "Morpho" pinout headers. Some of the morpho pin headers mapped the same as the Arduino female headers to allow the NUCLEO boards to debug when using shields. After programmed, the debugger can be pulled off and have a very compact microcontroller board. Then, the board can be programmed and debugged by attaching external cables back to the NUCLEO panel from the debugging PCB board.

In the ARM Cortex-M microcontroller community, the Serial Wire Debug protocol is entirely new. Instead of regular five-wire JTAG, which is more common on other boards, it only needed two wires.[14]

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 2.1.3.1 Technical Features

**Table 2.1 Comparison of NUCLEO and Arduino Boards [14].**

| | **NUCLEO F411RE** | **ARDUINO UNO** |
|---|---|---|
| **MICROCONTROLLER** | STM32F411RE 32-bit | ATMega 328 8-bit |
| **FAMILY** | ARM Cortex M4 | AVR |
| **CLOCK FREQUENCY** | 84 MHz | 16 MHz |
| **FLASH MEMORY** | 512 Kb | 32 Kb |
| **SRAM** | 96 Kb | 2 Kb |
| **EEPROM MEMORY** | - | 1 Kb |
| **PWM** | 10 | 6 |
| **ANALOG INPUTS** | 16 | 6 |
| **DIGITAL PIN** | 47 | 14 |
| **I2C MODULES** | 3 | 1 |
| **USART MODULES** | 3 | 1 |
| **SPI MODULES** | 4 | 1 |

| | | |
|---|---|---|
| **TIMER** | 10 | 3 |
| **FLOATING POINT UNIT** | One | N/A |
| **VOLTAGE (MAX)** | 5V | 5V |
| **USB OTG** | One | N/A |
| **DIMENSIONS** | 68x80mm | 53x68mm |
| **PRICE** | $14 | $25 |

From the table above, Arduino Uno and Nucleo Boards have a significant difference in performance, quality, input/output, and debugging options available in the market nowadays. However, the Nucleo board does not build-in internal EEPROM, which does not allow the permanent store of the variables in case the system is a reboot. The Arduino has a build-in EEPROM that allows the system to store the variables even when the system is a reboot.

In terms of performance, it surely can be observed that the NUCLEO processed with more complex algorithms. Furthermore, the C compiler in just a few instructions that allow faster execution and a distinct increase in performance. [14]

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

From the paper written by a student from the Department of Electronics and Communication Engineering, Gujarat Technological University, Ahmedabad, India, who is L. Louis, described the operating theory and implementations of the Arduino board[15]. Besides, he provides guidelines on how to apply Arduino as a tool for study and research purposes in this paper. This paper gives an insight into the form of Arduino boards, operating principles, implementation of code, and their applications. From this paper, I have studied the working principle of Arduino, its hardware/software features and applications, and where everything used at this moment. I have also learned how to write Arduino sketches in our IDE (software).

From the paper written by C. Rajan, B. Megala, A. Nandhini, et al. with the title of ‘A Review: Comparative Analysis of Arduino Micro Controllers in Robotic Car.’ It described that they did the operation of a mobile phone-controlled robotic car (remotely operated vehicle) that controlled by mobile phone (communicate over vast distances, even from various cities). The individual makes a telephone call to his mobile phone. When the call pressed, a tone corresponding to the pressed button is heard at the other end of the request if the button pressed. This tone is called the multiple frequencies of dual-tone DTMF. With the help of the phone stacked in the car, the car recognizes this DTMF ton. Arduino's microcontroller processes the received tone. The microcontroller is programmed to decide for any specific input and react towards the data given by either control the direction forward or backward or left or right direction. The mobile phone calling the cell phone in the car is functioning as a remote device. From this paper, I had learned the advantages of using the Arduino board, which are: Arduino Leonardo eliminates the necessity of secondary processor and Arduino Due used in complex projects without being easily replaceable. Arduino micro-supports quicker prototyping, wearable, and electronic materials. Lily pad Arduino, Arduino ESPRO-is equipped with joysticks, microphone, input side sensors, and output bumper, Arduino Yun -support cloud-based services, Arduino robots support our hardware parts. This study offers an extensive description of Arduino processors; many robot researchers will find it helpful.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

To have an idea to enchase the methodology in this project, I have also researched the comparative study of a different microcontroller. The paper, which title of ‘Overview and Comparative Study of Different Microcontrollers’ that written by R. Khadse, N. Gawai, and B. M. Faruk, described the essential information and comparative study of 8051 Microcontroller, ARM Microcontroller, PIC

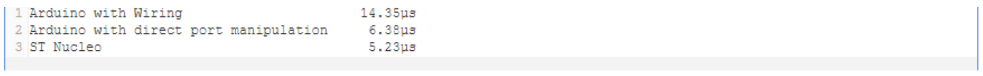
Microcontroller, AVR Microcontroller. The method that they used classified as the number of bits according to the microcontroller, architecture of instruction set, memory device last but not least, the classification of memory architecture[4]. From this classification, they form a method to identify the suitable comparison analysis study for them to determine which microcontroller is the best in performance.

Besides that, for me to have an idea on what suitable parameters that can be a standard benchmark for me to determine the system performance of a microcontroller, I had researched a paper which title is ‘An evaluation of energy-efficient microcontrollers’ that written by I. Tsekoura, G. Rebel, P. Glosekotter, and M. Berekovic[16]. In this paper, the writer described that in many projects, such as the Internet of Things, state-of-the-art technological developments need highly influential solutions. The low current power of the microcontrollers used to implement these solutions. However, it is harder than it seems to be to select the right microcontroller. The data sheets provide inadequate information to prevent fair comparisons between the different microcontrollers. This work attempts to solve this issue by using their laboratory's set of low-power microcontrollers and a range of benchmarks that perform common tasks. In terms of time, energy consumption, and energy consumption, they presented the results of a comparison among these microcontrollers in the performance of benchmarks. From this paper, I learned that as the operating frequency increases, power consumption increases, and the 32-bit microcontrollers have higher power consumption than 8-bit and 16-bit. Z8 Encore! Z8 Encore! XP has unusual behavior, as the microcontrollers in our List have the most significant energy consumption. But the MSP430 consumes less energy than the STM32F at smaller frequencies, an outstanding response for a 16-bit microcontroller. Lastly, the microcontrollers EFM32 and STM32F were the most efficient on our list about the computing performance per Joule.

### 2.1.3.2 Comparison of Minimum latency and timing of integer math operations.

From the paper that wrote by A. Ng, J. Lepinski, D. Wigdor, S. Sanders, and P. Dietz, with the title “Designing for Low-Latency Direct-Touch Input.” He described that latency is the period between the user's action and the device's reaction to that activity is inherent in every microcontroller; input sensors should be measured, computed, graphics produced, and displays modified. Existing company touch-screen applications have latencies ranging from 50 to 200 $ms$[17]. Ideally, device designers should retain this delay beneath. Threshold detectable by the human visual system, making the experience indistinguishable from a system that is genuinely latency-free. However, despite massive improvements in the efficiency of computer systems, latency remains an ever-present blemish on user experience[18]. According to the research of timing of integer, math operation is considered as an operation with purposely to determine the speed responsiveness of microcontroller to perform the arithmetic operations. With the research study, the author uses the microcontrollers Arduino UNO and ST NUCLEO L152RE to complete the survey. The study aims to compare the performance of these microcontrollers. Two experiments were conducted by the author, which are minimum latency measurement and integer math comparison. For the minimum latency measurement, the author determined that the ST NUCLEO L152RE ($5.23\mu s$) have the fastest speed than the Arduino UNO ($6.38\mu s$). The result is show in Figure 2.4 below:

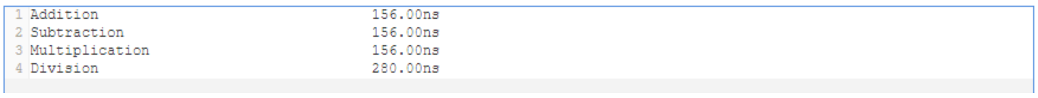
UNIVERSITI TEKNIKAL MALAYSIA MELAKA



```
Arduino with Wiring                    14.35µs
Arduino with direct port manipulation   6.38µs
ST Nucleo                               5.23µs
```

**Figure 2.4 Result of minimum latency measurement**[19]**.**

While for the integer math comparison, the author used the exactly similar integer math operations even with the same sequences math operations, the result has shown that the ST NUCLEO L152RE faster than the Arduino UNO. The result is shown in Figure 2.5 below:



```
Addition                        156.00ns
Subtraction                     156.00ns
Multiplication                  156.00ns
Division                        280.00ns
```

**Figure 2.5 The result of integer math comparison**[19].

The author used the oscilloscope to determine the time measurement for the whole experiment—each rising and falling edges of the waveform considered as the beginning and the end of the operations. Hence, measurement is taken at the rising and falling edges of the waveform. Thus, this method is much more exact or accurate to measure the time responsiveness towards these two experiments. Figure 2.6 below show the output waveform and the measurement of execution of time during the performance of integer math comparison by the author.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

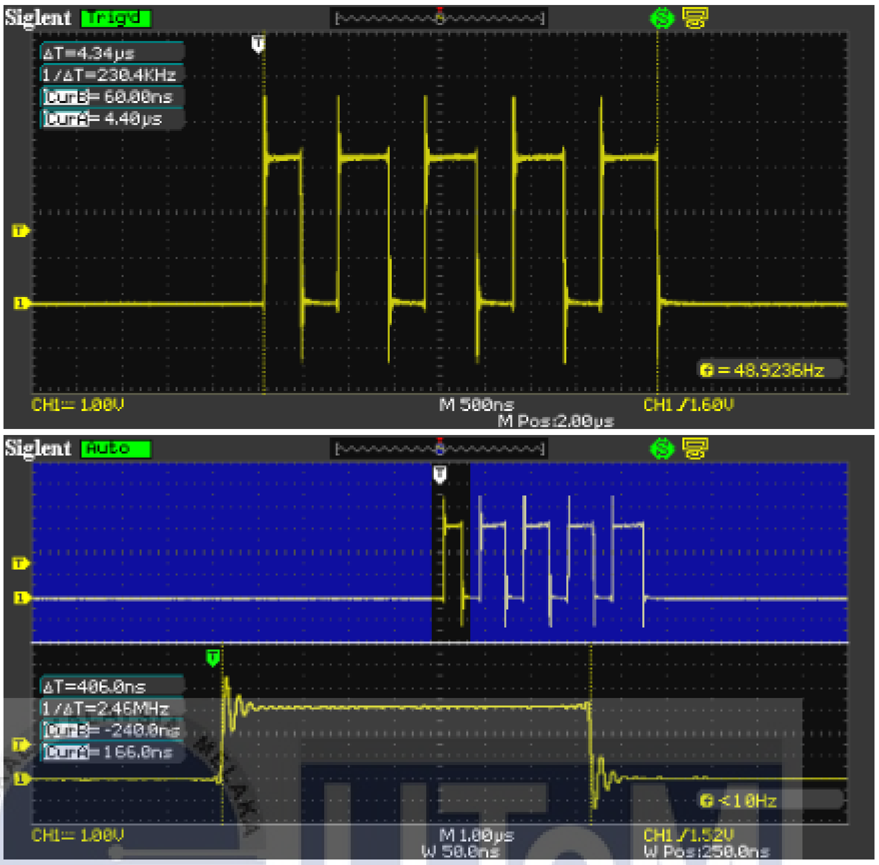


**Figure 2.6 The output waveform of execution time (integer math operation).**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 2.2 Water Monitoring System

For the protection of lives, water is a crucial element. Almost 70% of the world's surface water, but only about 2% of the total volume is freshwater for use and consumption. While population growth has risen at an unprecedented rate, the global community has only started to deal with the brunt of water shortages. It has only been heightened by unchecked urbanization and industrialization Currently. Freshwater resource quality and availability represent the most significant global problems of the world.[9]

The conditions of the swimming pool are directly dependent upon the control of its chemical characteristics [10]. The analysis of specific samples includes more rigorous testing is carried out manually by a human being. Therefore, the implementation of a sensor network that can manage these tests properly and accurately is necessary. The proposed system, the main objective is to automatically regulate and monitor swimming pools, the machine space condition, and the surrounding water quality. It designed to enhance its daytime maintenance rather than manually examine its water properties (pH, chlorine, water level, temperature).[20]

In this project, I will develop the smart water monitoring system to present a measurement that verifies and monitor the water quality with the given water sample using the wireless sensor network. Besides, it focuses primarily on the monitoring and data collection from node sensors, data storage to PMS databases, and display of historical and real-time data.

اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.2.1 pH meter (SKU: SEN0161)

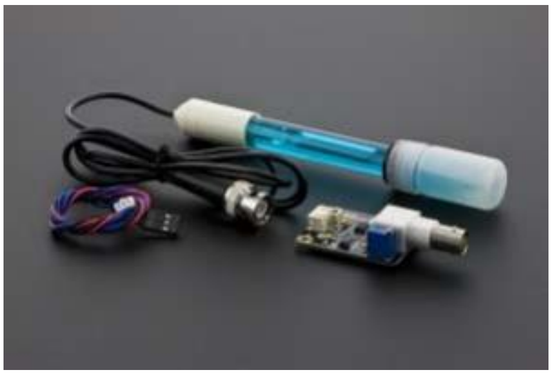


**Figure 2.7 Analog pH Meter Kit SKU: SEN0161.**

The DFRobot Gravity: Analog pH Meter Kit SKU: SEN0161 is to measure the water/solution pH level and deflect the acidity and alkalinity. They widely used in applications such as aquaponics, aquaculture, and environmental water monitoring system. It is compatible with Arduino and STM32 microcontroller boards. It has a built-in simple, convenient, and practical connection and features. Besides, LED is an indicator for Power indicator, a BNC connector, and a PH2.0 sensor interface. It is also simple to be installed and used by simply connect the pH sensor with BNC connector, and plug the PH2.0 interface into any analog input on a microcontroller to read pH value.[21]

**Table 2.2 pH Electrode Characteristics**

| Voltage(mV) | pH Value | Voltage(mV) | pH Value |
|---|---|---|---|
| 414.12 | 0.00 | -414.12 | 14.00 |
| 4354.96 | 1.00 | -354.96 | 13.00 |
| 295.80 | 2.00 | -295.80 | 12.00 |
| 236.64 | 3.00 | -236.64 | 11.00 |
| 177.48 | 4.00 | -177.48 | 10.00 |
| 118.32 | 5.00 | -118.32 | 9.00 |
| 59.16 | 6.00 | -59.16 | 8.00 |
| 0.00 | 7.00 | 0.00 | 7.00 |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.2.2 DS18B20 (Temperature Sensor)

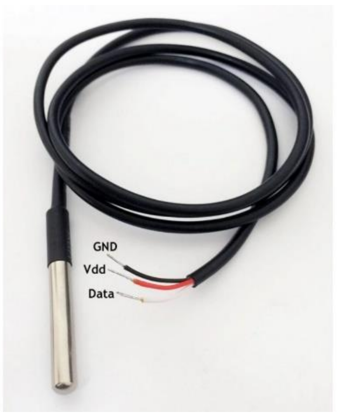


**Figure 2.8 Analogue temperature sensor (DS18B20).**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

The Electronic Temperature Sensor DS18B20 is fitted with a 9-bit to 12-bit temperature measurement and has an indicator feature with the upper and lower trigger points that used non-volatile. The DS18B20 communicates through a 1-wire bus that includes a single data line (and base) for interaction with a central microprocessor by default. The DS18B20 can also draw electricity directly from the data line and eliminate the need for an external power supply.

Each DS18B20 has a specific 64-bit serial protocol to operate on the same single1-wire bus. So, it is easy to use a microprocessor for controlling a considerable number of DS18B20s.

The implementations benefiting from this functionality include HVAC environmental control, within buildings, facilities, or machine tools, temperature control systems, and process surveillance and control systems[22].

**Table 2.3 Temperature/Data Relationship**

| TEMPERATURE (°C) | DIGITAL OUTPUT (BINARY) | DIGITAL OUTPUT (HEX) |
|---|---|---|
| +125 | 0000 0111 1101 0000 | 07D0h |
| +85 | 0000 0101 0101 0000 | 0550h |
| +25.0625 | 0000 0001 1001 0001 | 0191h |
| +10.125 | 0000 0000 1010 0010 | 00A2h |
| +0.5 | 0000 0000 0000 1000 | 0008h |
| 0 | 0000 0000 0000 0000 | 0000h |
| -0.5 | 1111 1111 1111 1000 | FFF8h |
| -10.125 | 1111 1111 0101 1110 | FF5Eh |
| -25.0625 | 1111 1110 0110 1111 | FE6Fh |
| -55 | 1111 1100 1001 0000 | FC90h |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.2.3 Water Quality Benchmark

Water quality chooses the 'goodness' of water for specific purposes. Water quality tests will provide data around the well-being of the conduit. By testing water over a while, the changes interior the quality of the stream seen. Parameters that endeavored connect temperature, pH, turbidity, saltiness, nitrates, and phosphates. An appraisal of the sea-going macro-invertebrates can identify other than provide a sign of water quality[23]. Besides, Water quality chose by the physical, chemical, and microbiological properties of water. These water quality characteristics all through the world characterized by wide changeability. Along these lines, the quality of conventional water sources utilized for specific purposes needs to build up in terms of the particular water-quality parameters that most impact the conceivable use of water[24].

#### 2.2.3.1 Temperature

The temperature of a conduit is essential since it impacts the total of broken-down oxygen interior the water. The whole of oxygen that will break up in water increases as temperature reduces. Water at 0℃ will hold up to 14.6 $mg$ of oxygen per liter, while at 30℃, it'll hold since it was up to 7.6 $mg/L$. Life cycles of sea-going life shapes are as frequently as conceivable related to changes in temperature. Temperature ranges for plants and animals influenced by artificial structures such as dams and weirs and releases of water from them[23].

When the precise estimation is required, we ought to continuously consider the temperature—the increment in temperature of water increments the ionization rate. As a result, the result appeared at 10°C is distinctive than at 25°C with the same esteem of pH and turbidity. Temperature plays a crucial part when measuring water quality. For case, pH esteem, as well as turbidity, changes with the alter in Temperature. pH is temperature subordinate, when the temperature goes up, the rate of ionization increments and bad habit versa. As an increase in temperature in arrangements, it causes diminish thickness and increment in the portability of particles in that arrangement. Since temperature is a fundamental portion of the changes in the values of materials within the water, the temperature sensor utilized. Detecting the temperature is basic. Heat is a critical component for deciding many other applications for water quality investigation[25][26].

Speed and response of the pH anode can different from the modify in temperature. Temperature can as well have impacts on the Calibration isothermal point, thermal

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

equilibrium, analytical balance, etc. At the common operations with the change in temperature, the result may adjust a little in the Acid region, while it can be significant in the Alkaline area.

**Table 2.4 Typical changes in pH for solutions due to temperature coefficient of variation effects**[27]**.**

| pH Range | Temperature | | |
|---|---|---|---|
| | 0℃ | 25℃ | 60℃ |
| Acid | pH 0.99 | pH 1.0 | pH 1.01 |
| Neutral | pH 7.47 | pH 7.00 | pH 6.51 |
| Alkaline | pH 14.94 | pH 14.00 | pH 13.02 |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Table 2.4 appears up the collections of pH respect with the alter in temperature. Acid incorporates a slight combination interior the pH connection as the temperature changes to 25℃ and 60℃ than at 0℃. At the fair-minded respect of pH, it has got to a few degrees more assortments as compared to the Acid. Though bases, it wraps an extraordinary collection with adjust in temperature between 0℃, 25℃, and 60℃ with the alter of pH respect of around 2.

#### 2.2.3.2 power of Hydrogen (pH)

PH is a measure of water acidity or alkaline. It is usually test-with decreased acidity or alkalinity with the color paper varies. Thanks to photosynthesis, pH typically fluctuates throughout streams. There are several explanations about why water may have strong pH values[23]. The pH value measured on a scale of 0 to 14, whereas non-water solutions will have a value of more or less than this. Neutral water has a pH value of 7.0 at ambient temperature of25℃, whereas solutions with a pH value of less than 7 are acidic and solutions with a pH value of more than 7 are alkaline. The logarithmic pH scale used to monitor and represent a broad spectrum of ion activity[26].

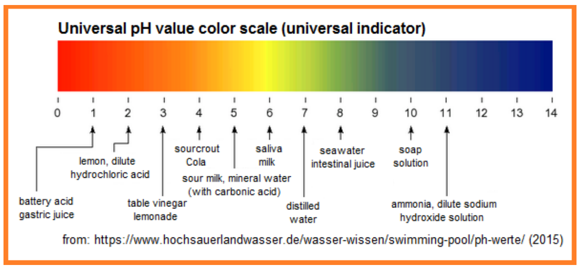


**Figure 2.9 Universal pH indicator** [28]**.**

Figure 2.6 indicates the thickness of the standard pH unit. Litmus paper is well known to be the most widely used form of necessary pH testing. Figure 2.6 shows the disparity in the color of litmus paper at various pH levels. The proper natural water pH level should be between 6.5 and 8.5, whereas the pH range for groundwater structures should be between 6 and 8.5.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.2.4 Review of implementation Smart Water Monitoring System with the Internet of Thing (IoT)

Form the paper that wrote by J. Gowthamy, C. R. Reddy, P. Meher, S. Shrivastava, and G. Kumar with the title "Smart Water Monitoring System using IoT [29]. He described that they did the research is due to assure secure water for various purposes such as agricultural consumption and usable water; the water regulated. According to this paper, the researcher had introduced a prototype of a minimal-cost water quality and water quantity monitoring system for IoT (internet of things). As a controller, the

Arduino model used. Finally, the sensor data sent through WI-FI via the internet. The data saving and review designed to include a cloud database so that these data used in future research and development.

Besides, written by K. Karimi and K. Salah-ddine and the title is “a Comparative Study of the Implementations Design for Smart Homes / Smart Phones Systems [30]. In this paper, it described that in reality, most people are using intelligent home technology to change home comfort and quality of life. Yet the option of the right one can be frustrating, given the numerous technical components. This paper provides a comparative study of critical elements such as communication protocols (Bluetooth, Wi-fi, Zigbee, and UWB), and the microcontrollers (Arduino and Pi Raspberry) and equipment of a Smart Home + smartphone network. It also outlines its main features and performances, such as power consumption, transmission time, and complexity.

Paper with the title “Water quality testing and monitoring system” written by N. Thirupathi Rao, D. Bhattacharyya, V. Madhusudhan Rao, and T. H. Kim [31]. They had described specimens that can be checked manually at the laboratory by using conventional methodologies for drinking water quality parameters, such as turbidity, pH, conductivity, temperature, etc. It solved by an attempt to develop the smart, low-cost IoT platform in the current article. Temperature, turbidity, pH, conduction are the criteria to determine water quality. The sensed sensor data transferred to the Raspberry Pi Unit, and the parameters then compared to the generic Raspberry Pi unit values. Link to data from IoT (cloud) in Raspberry Pi. If a change has occurred in the generic values, notification, or email sent via Wi-Fi to the smartphone, samples of water collected for water pureness tests in the current work, the different particles contained in the liquid also need to be tested. The current model will identify the particles

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

existing in the liquid and even the scale of pureness in the water. The findings presented in the form of numerical values at the display unit, which fitted to the IoT unit. Besides, the sensors used for testing water purity.

Last but not least, to further deepen understanding of a water monitoring system that implemented in my project, paper that wrote by M. Kumar Jha, R. Kumari Sah, M. S. Rashmitha, R. Sinha, B. Sujatha, and K. V. Suma. Title is “Smart Water Monitoring System for Real-Time Water Quality and Usage Monitoring,” which has studied [9]. From this paper, I learned how to develop of the SWMS in the area of in-house water quality control and use monitoring in real-time. From there, it inspired the ideas on improving the monitoring system in my project. There are two parts: Smart Water Quantity meter and Smart Water Quality Measurement. The goal, reporting this to the customer and authority by tracking the quantity of water consumed by a household and is the development of Smart Water Quantity Meter. A billing system of three plates produces bills of sale according to consumed amounts. By measuring five qualitative parameters, the Smart Water Quality meter verifies the purity of mobile water the consumer receives. By pH, temperature, turbidity, oxygen, and conductivity dissolved. The system ensures that any health hazards or possible threats caused by unintended leakage or discharge into portable water avoided. The Cloud data is supplied in real-time by an online surveillance system. Any violations of the use limit or water quality shall immediately be notified by SMS and a system-generated alert signal to the customer and authority.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Paper that wrote by A. N. Prasad, K. A. Mamun, F. R. Islam, and H. Haqva which title is “Smart water quality monitoring system,” has presented a smart water quality monitoring system for Fiji, using IoT and remote sensing technology[32]. The author simultaneously tested four different water samples to determine the parameters for each water sample. The process of the project carried within the indoor ambient temperature. It took 12 hours to complete the project with every 1-hour interval, and the readings recorded. To minimize or eliminate the error that could be affected by the project, the author kept the collected water samples and tested it in a safe, controlled environment. The water samples were changed every hour to keep the consistency readings. The result of this project matched the expected results obtained through their research. They also did succeed in implementing the sensors of potential Hydrogen (pH), temperature, GSM, conductivity and Oxidation, and Reduction Potential (ORP). From the result, the relation of temperature with pH and conductivity observed. Finally, all the results they obtained analyzed in the form of Neural Network Analysis[33].

The paper that wrote by S. Geetha and S. Gouthami, with the title of "Internet of things enabled real-time water quality monitoring system," had comprehensively described their project works in the field of water quality monitoring. Their research had also involved in power efficiency, solution for in-pipe water quality monitoring based on Internet of Things (IoT) technology[34]. They had developed a prototype that could determine and collect the water parameters, included the readings, and upload them to the cloud. The readings are then analyzed. Based on the analyzed result, the device alerted the remote user when there are any changes occurred, which differ from the pre-defined set of standard values. From this paper, it mentioned the proper way to test the water sample which all the sensors placed in the water. Then the sensors converted the physical parameters into a measurable electrical quantity, which given as input to the controller through an optional wireless communication device[34]. The whole project is used as a controller to read the data from the sensor,

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

process it, and sent the data to the application via communication technology. The parameters and communication devices can choose to monitored, which depends on the requirement of the use. This paper shows the proper methods (step-by-step) to develop the prototype and also the techniques to implement it into applications as well as the communication technology and Internet of Things (IoT-Ubidots). This paper used five parameters, which are conductivity, pH, turbidity, temperature, and water level. The IoT platform that used is Ubidots. The measured results compared with drinking water quality standards defined by WHO[34]. In conclusion form, this paper provides a comprehensive survey on the tools and methods implemented in existing smart water quality monitoring systems. It also provides techniques on how sensors communicate online to provides real-time online data to the user.

## 2.3 Wi-Fi Microcontroller

Wi-Fi microcontrollers allow Wi-Fi for devices to transmit and receive data and take orders. As a result, Wi-Fi microcontrollers used to carry ordinary appliances into the Internet.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.3.1 ESP8266

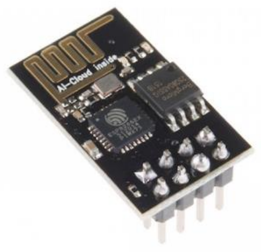


**Figure 2.10 ESP8266 Serial Wi-Fi Module.**

ESP8266EX is a highly integrated Wi-Fi SoC system designed to meet the continuous demands of consumers for efficient power usage, lightweight architecture and robust quality in the Internet of Things industry. For complete and self-contained Wi-Fi networking capabilities, ESP8266EX can function either as a standalone program or as a slave to the MCU host. When the program is powered by ESP8266EX, it will quickly boot up from the disk. The built-in high-speed cache helps increase system performance and maximize machine storage. ESP8266EX can also be added to any microcontroller model as a Wi-Fi adapter via SPI / SDIO or UART interfaces. The ESP8266EX incorporates antenna switches, RF Balun, control amplifier and low noise receives amplifier and filters, as well as power management modules.[35]

### 2.3.2 X-NUCLEO-IDW01M1

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

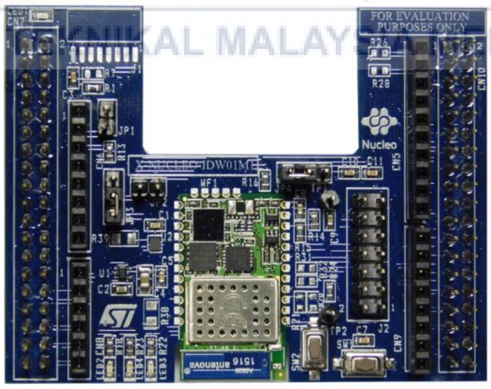


**Figure 2.11 X-CUCLEO-IDW01M1.**

The STM32 Nucleo boards expanded through the X-NUCLEO-IDW01M1 Wi-Fi evaluation board based on the SPWF01SA unit. STM32 MCU, Wi-Fi, and an SoC with integrated power amplifier and control as well as an SMD antenna are all included in the SPWF01SA certified CE, IC, and FCC unit. SPWF01SA also has an optional FLASH unit for software over-the-air upgrade (FOTA) of one MegaByte. The firmware has a full application IP stack that allows up to eight TCP / UDP ports, as well as the flexible SSI webpages to communicate with the device, and a REST API to pass data from/to the cloud server. The device will function as a socket server and socket user at the same time. The firmware supports stable TLS / SSL security sockets to allow safe end-to-end contact with or without authentication with the network. The unit used as STA, IBSS, or mini AP consumer (up to 5 STA customers). X-NUCLEO-IDW01M1 users can access the stack functions with the AT command on the STM32 Nucleo deck. It simplified via the UART serial connection. The ST morpho and the Arduino UNO R3 connector configuration are compliant with X-NUCLEO-IDW01M1.[36]

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 2.4 Internet of Things (IoT)

The internet is a global network system that uses satellite, database, mobile, network, and wireless network tools, and protocols, to allow communications between electronic devices. Two electronic gadgets can transmit data in many ways, such as document, voice, image, chart, and software, when linked via the internet.

In terms of its transmission rate, transmission approach, and implementation, Internet's capacity to develop and evolve is unprecedented. The rapid growth of the Internet and the evolution of information have intensified the need to implement new and innovative approaches that are accessible to meet customer needs. The Internet used to seek answers to existing problems or increasing current product features. This principle used by the Internet of Things (IoT) and has demonstrated tremendous success in a variety of applications such as consumer, commercial, industrial, and infrastructure sectors.

**Figure 2.12: Various applications of IoT** [37].

UTeM UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 2.4.1 Definition of Internet of Things (IoT)

The Internet of Things has no specific meaning, which is justifiable to community users worldwide. The term characterized by a wide range of groups, including academics, analysts, experts, researchers, technicians, and business persons. In 1999, Kevin Ashton, a specialist in digital innovation, created the earlier use of such a term "Internet of Things"[26]. He did not state the concept at that time but gave a clear view of IoT.

The same word or principle applies to many of the IoT meanings. The fundamental concept of the IoT is the interaction between two entities, which made up of a physical object and the internet. In a list below, the past studies show a particular IoT definition.

1. In a journal paper on the concept, challenges, and recent IoT studies, the researchers claimed that IoT is an inter-operable and autonomous, globally dynamic network structure. The authors also noted, however, that the meaning of IoT may differ depending on the context or listener [27].
2. S. Leminen et al. defined IoT as the ability to connect the Internet to all of the objects that surround us to allow everyone to access the data at any time and anywhere [28].
3. D. Singh et al. identified IoT as ' the Internet's things ' for offering and obtaining all data in the real world [29].

## 2.4.2 Components of Internet of Things (IoT)

IoT network consists of interconnecting parts such as sensors/devices, gateways, cloud, analysis, and user interfaces. The idea of the IoT is to communicate multiple devices and objects that are capable of transferring information and of making the necessary decisions that eliminate interactions between humans and human beings.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

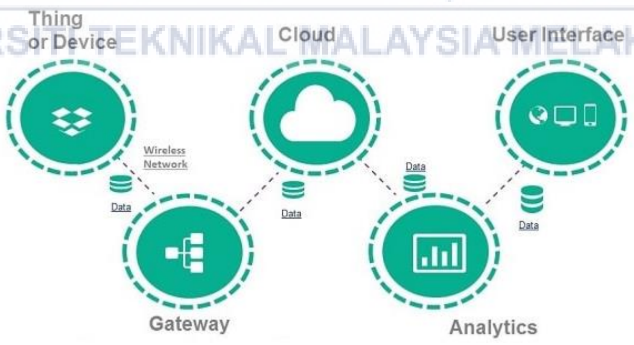


**Figure 2.13: Major components of IoT.**

The first phase in the IoT network is a collection of data. Sensors or devices have been used in first structures to continuously capture live data from the surroundings and transfer to the next processing step. Sensors and tools can vary with various IoT applications. Analog pH meter, temperature, etc. are the most often used sensors or devices. This project uses the water monitoring system to retrieve data from the water sample that provided.

The second element stage of the IoT system is the gateway for information processing and monitoring placed. IoT Gateway handles network data moves from the protocol to the network. Furthermore, it functions as a middle layer among both sensors and the cloud. In this venture, Arduino UNO R3 and STM32F411RE served as a gateway in the IoT network.

In IoT, massive amounts of data from sensors or computers can be processed and analyzed. Cloud has a significant role to play in the collection, analysis, and storage of large amounts of IoT information in real-time. End users or consumers can reach this information only with internet access online at every location. Many IoT Cloud Platforms, as well as for open-source and paid, were designed to address IoT's enormous growth in various industries. Table 2.4 below shows some examples of IoT cloud platform and their vendor.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Table 2.5: List of IoT cloud platform** [38]**.**

| Vendor | Platform | Open Source or Paid? | Recognition and compliance |
|---|---|---|---|
| ioBridge | ThingSpeak | Open Source | - |
| CyberVision | Kaa | Open Source | - |
| | DeviceHive | Open Source | - |
| Particle.io | Particle | Open Source | - |
| Microsoft | Azure | Open Source | - |

| | | | |
|---|---|---|---|
| TheThings.io | TheThings | Paid | Forbes |
| Aeris | Aercloud | Paid | Forrester, SimplyHome wins Telehealth 2015 award |
| Ayla Networks | Ayla Networks | Paid | Gartner Cool Vendor 2015, Forbes |
| IBM | Bluemix | Paid | Forrester |
| LogMeIn.Inc | Xively | Paid | Forbes, Forrester, 6+ awards including World's top 10 innovative IoT company by Fast Company in 2014 |
| PTC | ThingsWorx | Paid | 10+ awards including Gartner cool vendor 2014, Safe Harbor Certified |
| Temboo.Inc | Temboo | Paid | Forbes |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Analytics means that multiple analog information from sensors and devices converted into an information-analysis format. The IoT platform introduced to work on real-time data analytics and respond to unusual events. The IoT system's analysis process is beneficial to users by showcasing the potential pattern based on the retrieved machine learning. It can also forecast outcomes on a trend-based basis so that users can take a step further. Therefore, ThingSpeak chose as the IoT platform in this project. Data analytics applied to the data from the water monitoring system. This water monitoring system will collect the data from the water sample that provided. It will upload the data to the cloud to the IoT platform to display the reading in the ThingSpeak IoT platform's dashboard. From there, the microcontrollers' system performance will be determined and analyzed.

# CHAPTER 3

# METHODOLOGY

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

This chapter explained the methods used and steps taken in gathering information, collecting data, and result evaluation to accomplish the objectives of this project. The system block diagram shows an overview of the project functionality. A flow chart illustrates the procedure flow from the beginning of the project until the end of it. The descriptions of each stage in the project flow chat are described clearly in this section. Methodology acts as a guideline in project management to ensure that all the works executed on the right track.

## 3.1 Overview of the project

According to the block diagram and flowchart, as shown in Figure 3.1 and Figure 3.2 respectively, the knowledge about the system performance test should be studied deeply to conduct and carry out experiments to test the system performance of the Arduino Uno and STM32F411RE microcontrollers board. After obtained the result, the microcontroller developed into a water monitoring system.

Speed performance tests, both of the microcontrollers (Arduino Uno and STM32F411RE), will be undergoing a speed test performance. This speed performance test divided into two experiments. These two experiments aim to test the minimum latency and the timings of the operations of the integer arithmetic process. The same and exact arithmetic operation sequences with RISC instruction set architecture will be used. The time to execute each of the instructions will be recorded then analyzed.

Finally, after obtained the speed performance test result, both of the microcontrollers will be developed into the smart water monitoring system. To ensure the objectivity to be achieved, both microcontrollers will be programmed by using RISC instructions set, same and exact with no other operations, and the same sensors used. The aim of this development of the Smart Water Monitoring System is firstly to compare and study the performance system of both different microcontrollers in different platforms. Secondly, to achieve the objective of this project and thirdly is to implement both microcontrollers into nowadays trending feature, which is the Industry Revolution 4.0 (IR4.0). Besides, the priority of this development is to choose the best and compatible sensors so both different platform microcontroller could use it. Hence, the SKU SEN0161 (analog PH meter) and DS18B20 (analog Temperature) chose in this project to measure the temperature and pH value reading.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

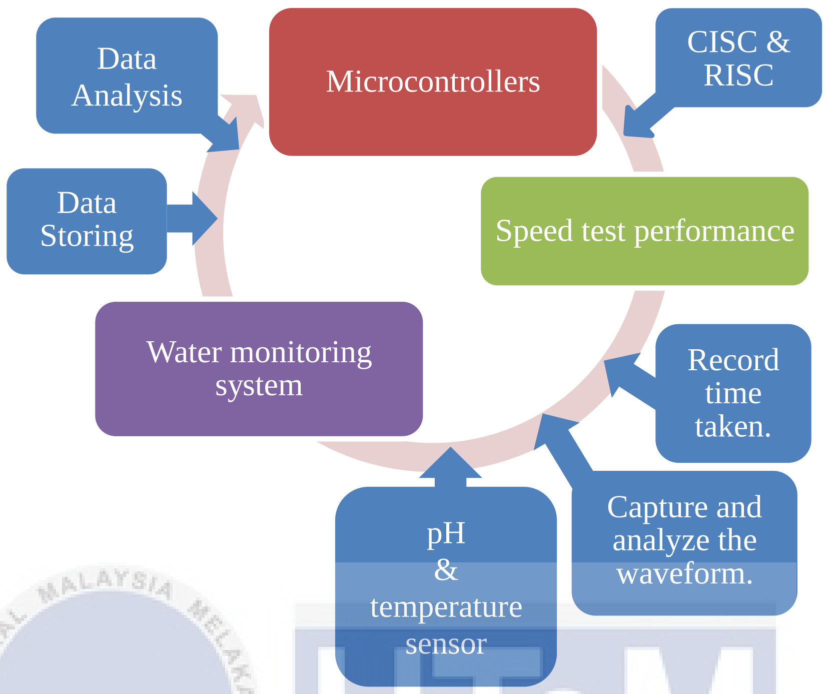


**Figure 3.1 Block diagram of the project.**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 3.1.1 System Performance Implementation

#### 3.1.1.1 Latency Test

Latency is an associate expression of what proportion time it takes for an information packet to transfer from one selected purpose to a different. Ideally, latency is going to be as about to zero as potential. For the experiment to test the minimum latency measurement, I set up an input pin as well as an output pin. The two pins are linked together via a wire, and an oscilloscope is attached—the input pin connected to an interrupt handler reacting with every edge. The aim is to set the output pin to the reverse pin value. Because the output is linked back to the input, which refers to an infinite collection of value changes generating a square wave of twice the minimum latency. With the square wave that generated, it observed from the oscilloscope. The start-stop time of the waveform measured as the time response of the microcontroller to react towards interrupt, which called latency.

Since both of the microcontrollers using a different platform, which means even though both are microcontroller but using slightly different program software, language, and syntax. Hence, in this test, different language and syntax codes to implement will be used to execute the same and exact operation, at the same time, achieve the objective that set in Chapter 1.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 3.1.1.2 Timings of the Operations of Integer Arithmetic Process

The integer arithmetic provided if the expression includes all the numbers, there will be no problem of subtraction, addition, multiplication, and exponentiation. With actual values, though, the integer division is very different from the usual division. The Integer division ignores the fractional portion. They're truncating both decimal marks. In this project, the system performance of both microcontrollers tested with the timely completion of the operation of the Integer Arithmetic Process. The oscilloscope will capture the waveform of the operation in this process. The start-stop of the waveform indicated as the begin and end of each arithmetic operations (addition, subtraction, multiplication, and division).

To measure the integer arithmetic process time, I configured both of the microcontrollers one pin as an output pin and increased the output before the operation and then lowered it. It will provide a burst for each operation, and the pulse length corresponds to the integer arithmetic process time, considering the time necessary for increasing and growing the output. The integer arithmetic operation was skipped in the first signal to offset for the time to raise and lower the output. The first pulse, therefore, provides the time to remove it from the other pulses to get the pure integer arithmetic process time. The successive pulses contain addition, subtraction, multiplication, and division.

### 3.1.2 Implementation of Code (Speed Performance Test)

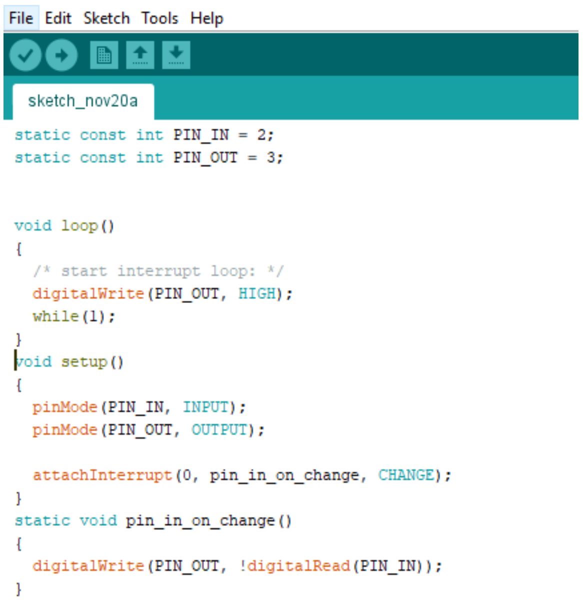
#### 3.1.2.1 Minimum Latency Test

A project to measure and test the minimum latency and integer math operation, one of the pins from the microcontroller, chose as input and another as an output. A wire linked both pins, and between the connection, the oscilloscope attached to this connection. Attached to the input pin was an interrupt handler that reacts on each edge. It intends to set an output pin to input pin's inverse value. It leads to an infinite sequence of value changes as the output is connected back to the origin, thereby producing a square wave with a period that is double the typical latency.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

##### *(a)* Arduino UNO R3

Since the output is related to the origin, this results in a continuous series of shifts in frequency, producing a square wave with a period that is double the usual duration. The input and the output set to pin two and pin three, respectively. The functionality of the wiring framework, as described on the Arduino website are used in this edition and is probably the best way most users of Arduino use it.

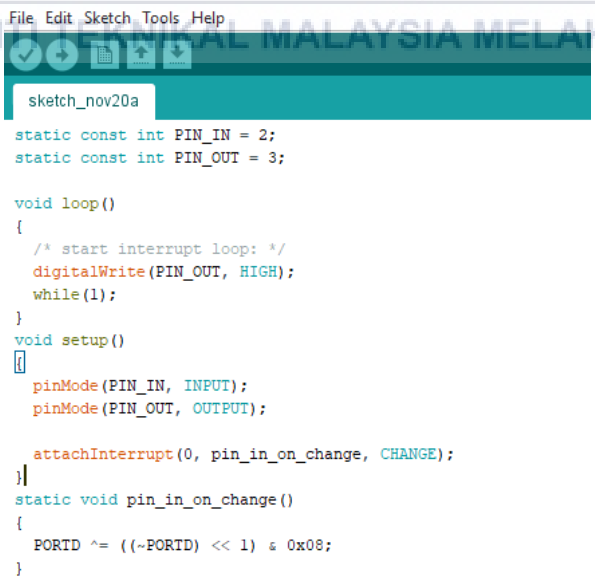


```
static const int PIN_IN = 2;
static const int PIN_OUT = 3;


void loop()
{
  /* start interrupt loop: */
  digitalWrite(PIN_OUT, HIGH);
  while(1);
}
void setup()
{
  pinMode(PIN_IN, INPUT);
  pinMode(PIN_OUT, OUTPUT);

  attachInterrupt(0, pin_in_on_change, CHANGE);
}
static void pin_in_on_change()
{
  digitalWrite(PIN_OUT, !digitalRead(PIN_IN));
}
```

**Figure 3.2 Arduino with Pure Wiring Functions code.**

Since there is an alternative way to implement the interrupt handler, which directly manipulates the port register that allows saving a lot of time, but it mostly used by advanced users.

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA



```
static const int PIN_IN = 2;
static const int PIN_OUT = 3;

void loop()
{
  /* start interrupt loop: */
  digitalWrite(PIN_OUT, HIGH);
  while(1);
}
void setup()
{
  pinMode(PIN_IN, INPUT);
  pinMode(PIN_OUT, OUTPUT);

  attachInterrupt(0, pin_in_on_change, CHANGE);
}
static void pin_in_on_change()
{
  PORTD ^= ((~PORTD) << 1) & 0x08;
}
```

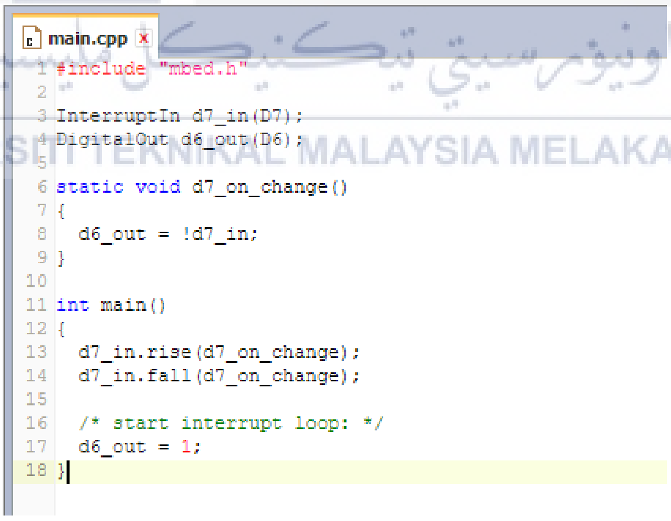
**Figure 3.3 Arduino with Direct Manipulation of Port Register code.**

The second iteration is a much efficient way to execute the interrupt handler, which only been used by more advanced people. The port register is directly regulated here, thus saving a lot of time. From there, the waveform measured and obtained through the oscilloscope.

*(b)* **STM32F411RE**

With STM32F411RE, the configuration pin of this board is almost similar to Arduino UNO R3. Hence, to make it standardize, the same pin that declared or used in Arduino UNO R3 in the same test will be used in STM32F411RE. Pin 6 and 7 used as input and output, respectively. The same equipment (oscilloscope) used to measure and obtained the waveform generated. Next, to identify the time is taken, and the period of the execution time required, analysis of that waveform done.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



```
#include "mbed.h"

InterruptIn d7_in(D7);
DigitalOut d6_out(D6);

static void d7_on_change()
{
  d6_out = !d7_in;
}

int main()
{
  d7_in.rise(d7_on_change);
  d7_in.fall(d7_on_change);

  /* start interrupt loop: */
  d6_out = 1;
}
```

**Figure 3.4 STM32F411RE Latency Test Functions code.**

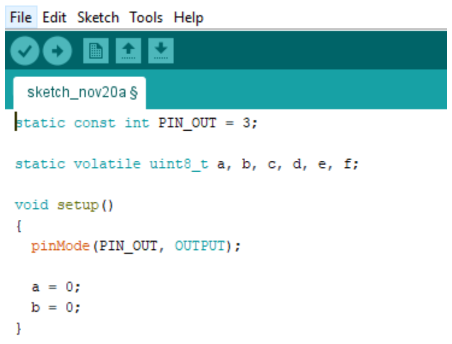
### 3.1.2.2 Timings of the Operations of Integer Arithmetic Process

For testing the timing of the timings of the Operations of Integer Arithmetic Process, I had set the pin three as an output pin for both microcontrollers. To observe the significant difference of timings of the Operations of Integer Arithmetic Process, at first, I run the test by using the Arduino UNO R3 with the 8-bit arithmetic operation and obtained the time taken for each operation. Then, I increased the number of bits to became 16 and 32 bits. Hence, every waveform captured, and the time is taken or a period of execution of each arithmetic operation recorded.

#### *(a)* Arduino UNO R3

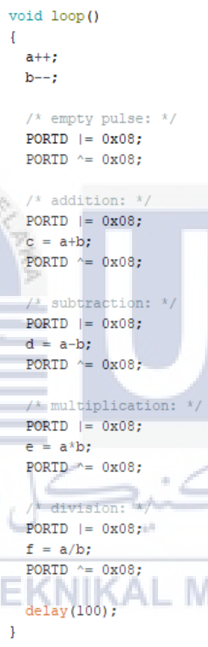
For Arduino arithmetic integer code, pin three assigned as the output pin, which will produce the output signal waveform into the waveform. To perform the arithmetic integer, a, b, c, d, e, and f assigned, which will begin with 8-bits, then increase to 16-bits and finally 32-bits. Initially, the ‘a’ and ‘b’ attached with 0, every looping, the ‘a’ and ‘b’ will increment by 1. The direct port assigned is being used. At first, the pin 3 (port D) will produce a high signal (1) which perform ‘OR’ gate operation. Then, pin three will become 0 that this time assigned to perform the ‘XOR’ gate operation. Hence, the square wave produced.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



```
static const int PIN_OUT = 3;

static volatile uint8_t a, b, c, d, e, f;

void setup()
{
  pinMode(PIN_OUT, OUTPUT);

  a = 0;
  b = 0;
}
```

For this, each arithmetic integer operation assigned the pin three as High output signal (1) then, after completed each of this arithmetic integer operation instruction, the pin three will be assigned to become the low output signal (0). Which indicated, the completed of the arithmetic integer operation. The output signal produced from the Arduino UNO R3 is then connected directly to the oscilloscope. Figure 3.4 in the below is the waveform produced by the microcontroller Arduino UNO R3 that implemented the arithmetic integer operation.



```
void loop()
{
  a++;
  b--;

  /* empty pulse: */
  PORTD |= 0x08;
  PORTD ^= 0x08;

  /* addition: */
  PORTD |= 0x08;
  c = a+b;
  PORTD ^= 0x08;

  /* subtraction: */
  PORTD |= 0x08;
  d = a-b;
  PORTD ^= 0x08;

  /* multiplication: */
  PORTD |= 0x08;
  e = a*b;
  PORTD ^= 0x08;

  /* division: */
  PORTD |= 0x08;
  f = a/b;
  PORTD ^= 0x08;

  delay(100);
}
```

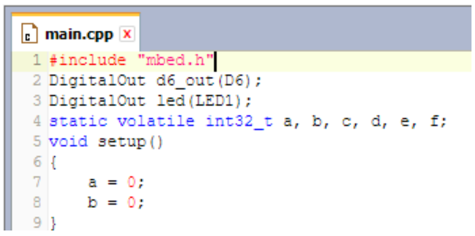
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Figure 3.5 Arduino UNO R3 integer arithmetic operations code.**

### *(b)* STM32F411RE

For STM32F411RE arithmetic integer code, pin six assigned as the output pin, which will produce the output signal waveform into the waveform. To perform the arithmetic integer, a, b, c, d, e, and f appointed, which will begin with 8-bits, then increase to 16-bits and finally 32-bits. Initially, the ‘a’ and ‘b’ assigned with 0, every looping, the ‘a’ and ‘b’ will increment by 1. The direct port assigned is being used.

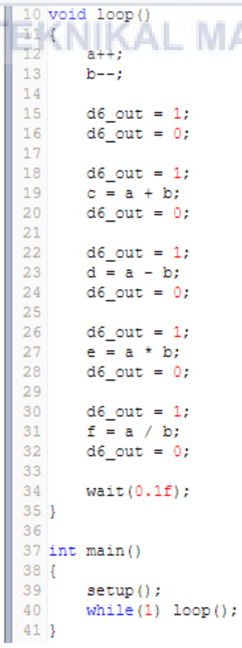
At first, the pin 3 (port D) will produce a high signal (1) which perform ‘OR’ gate operation. Then, pin three will become 0 that this time assigned to perform the ‘XOR’ gate operation. Hence, the square wave produced.



```
main.cpp
#include "mbed.h"
DigitalOut d6_out(D6);
DigitalOut led(LED1);
static volatile int32_t a, b, c, d, e, f;
void setup()
{
    a = 0;
    b = 0;
}
```

For this, each arithmetic integer operation assigned the pin three as High output signal (1) then, after completed each of this arithmetic integer operation instruction, the pin three will be assigned to become the low output signal (0). Which indicated, the completed of the arithmetic integer operation. The output signal produced from the STM32F411RE is then connected directly to the oscilloscope. Figure 3.5 in the below is the waveform produced by the microcontroller STM32F411RE that implemented the arithmetic integer operation.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



```
void loop()
{
    a++;
    b--;

    d6_out = 1;
    d6_out = 0;

    d6_out = 1;
    c = a + b;
    d6_out = 0;

    d6_out = 1;
    d = a - b;
    d6_out = 0;

    d6_out = 1;
    e = a * b;
    d6_out = 0;

    d6_out = 1;
    f = a / b;
    d6_out = 0;

    wait(0.1f);
}

int main()
{
    setup();
    while(1) loop();
}
```

**Figure 3.6 STM32F411RE integer arithmetic operations code.**

### 3.1.3 Water Monitoring System (Internet of Things)

For the implementation of the Water Monitoring System, I have written the code for assigned the two pins, which are A0 and A1 for the analog pH sensor and temperature sensor, respectively. A complete system by combining pH sensor, temperature sensor, and Wi-Fi module functions into the main function loop to extract solution data and send it to the cloud at the same time. The data request and data collection of the two parameters used in this project will proceed first. These data sensors will be further process by the mathematical formula based on the parameter, and data bytes received. After acquiring all sensor data, it sent to the cloud server via a Wi-Fi module. A delay of 15 seconds implemented at the end of the function loop because ThingSpeak requires 15 seconds of interval for the data update to its cloud server.

UTeM
اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 3.1.3.1 Arduino UNO R3

Firstly, pin A0 and A1 are declared as receiver of pH sensor and temperature. To program these two sensors, I used OneWire.h and DallasTemperature.h libraries for the temperature sensor (DS18B20). While for the analog pH sensor (SEN0161), DFRobot_PH.h, and EEPROM.h libraries. By using these libraries, all the functionality declared in .cpp and .h file. Code to program both sensors is coded in pH() and temperature() function for reading the pH value and temperature value, respectively. The figure below shows the code to program pH and temperature sensors.

```
void pH(void)//ontop of main()
{
  static unsigned long timepoint = millis();
    if(millis()-timepoint>1000U)  //time interval: 1s
    {
      timepoint = millis();
      voltage = analogRead(PH_PIN)/1024.0*5000;  // read the voltage
      float temp = 0;
      phValue = ph.readPH(voltage,temp);  // convert voltage to pH with temperature 
compensation
      Serial.print("^C  pH:");
      Serial.println(phValue,6);
    }
}

void temperature()
{
  sensors.requestTemperatures(); //request the temperature
  Celcius=sensors.getTempCByIndex(0);
  Serial.print(" C  ");
  Serial.print(Celcius);
}
```

the Serial monitor, ESP8266, pH sensor as well as the temperature sensors. To enable the Arduino UNO R3 to communicate with ESP8266, AT command is created to allow the communication. Command also includes the authority to connect to the Wi-Fi network. Once the connection has connected, it will print the “Connected” and “Water Monitoring System Demo” in the Serial monitor.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

```
void setup()
{
  Serial.begin(9600);
  esp.begin(9600);
  ph.begin();
  sensors.begin();

  send_command("AT+RST\r\n", 2000, DEBUG); //reset module
  send_command("AT+CWMODE=1\r\n", 1000, DEBUG); //set station mode
  send_command("AT+CWJAP=\"terry1\",\"123456789\"\r\n", 2000, DEBUG);   //connect wifi
network

while(!esp.find("OK"))
  { //wait for connection
  Serial.println("Connected/n");
  Serial.println("Water Monitoring System Demo/n");
  }
}
```

The figure below shows to send the data to ESP8266; function send_command created. The params that used are purposely: command - the data/command to send; timeout - the time to wait for a response; debug - print to Serial monitor? (true = yes, false = no); Returns: The response from the ESP8266 (if there is a response).

```
String send_command(String command, const int timeout, boolean debug)
{
  String response = "";
  esp.print(command); // send the read character to the esp8266
  long int time = millis();
  while ( (time + timeout) > millis())
  {
    while (esp.available())
    {
      // The esp has data so display its output to the serial window
      char c = esp.read();// read the next character.
      response += c;
    }
  }
  if (debug)
  {
    Serial.print(response);
  }
  return response;
}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

To upload the real-time data, update data() created. In this function, this is to ensure all the new data uploaded into the IoT platform by using an HTTP request. In this project, field one and field two used to upload the pH level data and temperature data, respectively. The figure below shows the coding to upload the data into ThingSpeak.

```
void updatedata()
{
  String command = "AT+CIPSTART=\"TCP\",\"";
  command += IP;
  command += "\",80";
  Serial.println(command);
  esp.println(command);
  Serial.print(" Requesting temperatures & pH...");
  Serial.print("  Temp Celcius: ");
  Serial.print(Celcius,2);
  Serial.print("     pH value: ");
  Serial.println(phValue,6);
if(esp.find("Error"))
{
    return;
}
command
command = Api_key ;
command += "&field1=";
command += phValue;
command += "\r\n";
command += "&field2=";
command += Celcius;
command += "\r\n";

Serial.print("AT+CIPSEND=");
esp.print("AT+CIPSEND=");
Serial.println(command.length());
esp.println(command.length());
if(esp.find(">"))
{
  Serial.print(command);
  esp.print(command);

}
else
{
 Serial.println("AT+CIPCLOSE");
 esp.println("AT+CIPCLOSE");
  //Resend...
  error=1;
}
}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

After successfully implement the Arduino code of each function block, a complete Arduino coding by combining these sensors' functions block and ESP8266 functions block into the main function loop to extract the sensor data and send it to the cloud at the same time. The complete coding below shown that the data request and data collection of two parameters used in this project will proceed first. These data will be further process by the mathematical formula based on the parameter, and data bytes received. After acquiring sensor data, it sent to the cloud server via ESP8266. A delay of 15 seconds implemented at the end of the function loop because ThingSpeak requires 15 seconds of interval for the data update to its cloud server.

```
#include <SoftwareSerial.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include "DFRobot_PH.h"
#include <EEPROM.h>

///////////////////ESP8266////////////////////
SoftwareSerial esp(2,3);
//////////////////PH/////////////////////////
#define PH_PIN A0
float voltage,phValue;
DFRobot_PH ph;
/////////////////Temperature//////////////////
#define tempSensor A1
OneWire oneWire(tempSensor);
DallasTemperature sensors(&oneWire);
```

```
float Celcius=0;
//////////////////Thingspeak///////////////////
#define DEBUG true
#define IP "184.106.153.149"// thingspeak.com ip
String Api_key = "GET /update?key=K1MN6S459Y4HL56K"; //change it with your api key like "GET
/update?key=Your Api Key"
int error;

void setup()
{
  Serial.begin(9600);
  esp.begin(9600);
  ph.begin();
  sensors.begin();

  send_command("AT+RST\r\n", 2000, DEBUG); //reset module
  send_command("AT+CWMODE=1\r\n", 1000, DEBUG); //set station mode
  send_command("AT+CWJAP=\"terry1\",\"123456789\"\r\n", 2000, DEBUG);   //connect wifi
network
  while(!esp.find("OK"))
  { //wait for connection
  Serial.println("Connected/n");
  Serial.println("Water Monitoring System Demo/n");
  }
}

void pH(void)//ontop of main()
{
  static unsigned long timepoint = millis();
    if(millis()-timepoint>1000U)  //time interval: 1s
    {
      timepoint = millis();
      voltage = analogRead(PH_PIN)/1024.0*5000;  // read the voltage
      float temp = 0;  // read your temperature sensor to execute temperature compensation
      phValue = ph.readPH(voltage,temp);  // convert voltage to pH with temperature
compensation
      Serial.print("^C  pH:");
      Serial.println(phValue,6);
    }
}

void temperature()
{
  sensors.requestTemperatures();
  Celcius=sensors.getTempCByIndex(0);
  Serial.print(" C  ");
  Serial.print(Celcius);
}

void loop()
{
  pH();
  temperature();
  start: //label
  error=0;
  updatedata();
  if (error==1)
  {
    goto start; //go to label "start"
  }
  delay(1000);
}

void updatedata()
{
  String command = "AT+CIPSTART=\"TCP\",\"";
  command += IP;
  command += "\",80";
  Serial.println(command);
  esp.println(command);
  Serial.print(" Requesting temperatures & pH...");
  Serial.print("  Temp Celcius: ");
  Serial.print(Celcius,2);
  Serial.print("    pH value: ");
  Serial.println(phValue,6);
  if(esp.find("Error"))
  {
    return;
  }
  command = Api_key ;
  command += "&field1=";
  command += phValue;
  command += "\r\n";
  command += "&field2=";
  command += Celcius;
  command += "\r\n";
```

UTeM
اونيۏرسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

```
  Serial.print("AT+CIPSEND=");
  esp.print("AT+CIPSEND=");
  Serial.println(command.length());
  esp.println(command.length());
  if(esp.find(">"))
  {
    Serial.print(command);
    esp.print(command);

  }
  else
  {
   Serial.println("AT+CIPCLOSE");
   esp.println("AT+CIPCLOSE");
    //Resend...
    error=1;
  }
  }

String send_command(String command, const int timeout, boolean debug)
{
  String response = "";
  esp.print(command);
  long int time = millis();
  while ( (time + timeout) > millis())
  {
    while (esp.available())
    {
      char c = esp.read();
      response += c;
    }
  }
  if (debug)
  {
    Serial.print(response);
  }
  return response;
}
```

### 3.1.3.2 STM32F411RE

Firstly, pin A0 and A1 are declared as receiver of pH sensor and temperature. To program these two sensors, I used DS1820.h libraries for the temperature sensor (DS18B20). While for the analog pH sensor (SEN0161), SEN0161.h libraries. By using these libraries, all the functionality declared in .cpp and .h file. Code to program both sensors is coded inside the main() functions block. The figure below shows the code written to program the sensors.

```
float voltage=sensor.get_pH();
float pH=-6*voltage+16;
printf("\n\r PH is %f\n", pH);
//float voltage= (sensor.get_pH())/3.5f;
//printf("\n\r mV is %f\n", voltage);
ds1820.startConversion();   // start temperature conversion
float temp=ds1820.read();
printf("\n\r temp = %3.1f\r\n", temp);     // read temperature
```

As the mbed compiler has the different coding format and syntax, the setup functions which usually wrote in the Arduino IDE compiler is now become the main() functions

UTeM UNIVERSITI TEKNIKAL MALAYSIA MELAKA اونيۏرسيتي تيكنيكل مليسيا ملاك

block. In this main() features block, the system begins with the setup of the Wi-Fi module. It provides with the Wi-Fi information, SSID, and password. If the module succeeds in connecting to the Wi-Fi, then the microcontroller will print the result “connected” else it will print “error connecting.” Once it is connected, the IP and Mac address of the Wi-Fi connection and microcontroller device’s address will be display on the Tera Term. The figure below shows the setup of the whole system in the mbed compiler.

```
int main()
{
    int err;
    char * ssid = "terry1";
    char * seckey = "123456789";
    pc.printf("\r\nNucleo F411RE + WiFi + Thingspeak\r\n");
    pc.printf("\r\nconnecting to AP\r\n");

    if(spwf.connect(ssid, seckey, NSAPI_SECURITY_WPA2)) {
        pc.printf("\r\nnow connected\r\n");
    } else {
        pc.printf("\r\nerror connecting to AP.\r\n");
        return -1;
    }

    const char *ip = spwf.get_ip_address();
    const char *mac = spwf.get_mac_address();

    pc.printf("\r\nIP Address is: %s\r\n", (ip) ? ip : "No IP");
    pc.printf("\r\nMAC Address is: %s\r\n", (mac) ? mac : "No MAC");
    pc.printf("\r\n--Starting--\r\n");

    while(1)
        {
            wait(1.0);
            myled = !myled;
         }
}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

To send the data to update the real-time data that obtained or captured by the sensors to the IoT platform, in the main() functions to block the code written as shown in the figure below to ensure each new data uploaded into the IoT platform.

```
int err=http_demo(pH,temp);

   if(err==0)
       pc.printf("Thingspeak update completed successfully\r\n");
   else
       pc.printf("Error occurred %d\r\n",err);
```

After update newly data, the data sent to the cloud via the IDW01M1 Wi-Fi module. To do so, code to program the IDW01M1 created in the http_demo() functions block, which shown in the figure below.

```
int http_demo(long pH, long temp)
{
    TCPSocket socket(&spwf);
    char buffer[512];
    char message[80];
    int err;
    printf("Sending HTTP Data to thingspeak...\r\n");

    // Open a socket on the network interface, and create a TCP connection to thingspeak
    //socket.open(&spwf);
    err=socket.connect(IP,80); // connecting to thingspeak
    if(err!=0) {
        pc.printf("\r\nCould not connect to Socket, err = %d!!\r\n", err);
        return -1;
        }
    else
        pc.printf("\r\nconnected to host server\r\n");

        // Send a simple http request
        sprintf(message,"field1=%d&field2=%d,pH,temp);

    printf("Message Length=%d\r\n",(int)strlen(message));
    sprintf(buffer,"POST    /update    HTTP/1.1\r\nHost:    api.thingspeak.com\r\nConnection:
close\r\nX-THINGSPEAKAPIKEY:            %s\r\nContent-Type:            application/x-www-form-
urlencoded\r\nContent-Length: %d\r\n\r\n%s",thingSpeakKey,(int)strlen(message),message);
    pc.printf("Request to %s\r\n", buffer);
    //char sbuffer[] = "GET / HTTP/1.1\r\nHost: www.arm.com\r\n\r\n";
    int scount = socket.send(buffer, (int)strlen(buffer));
    printf("sent %d [%.*s]\r\n", scount, strstr(buffer, "\r\n")-buffer, buffer);

    // Recieve a simple http response and print out the response line
    char rbuffer[64];
    int rcount = socket.recv(rbuffer, sizeof rbuffer);
    printf("recv %d [%.*s]\r\n", rcount, strstr(rbuffer, "\r\n")-rbuffer, rbuffer);

    // Close the socket to return its memory and bring down the network interface
    socket.close();
    return 0;
}
```

After successfully implementing the STM32F411RE code of each function block, a complete coding by combining these sensors' functions block and IDW01M1 functions block into the main function loop to extract the sensor data and send it to the cloud at the same time. In this project, the complete coding below shown that the data request and data collection of the two parameters used will proceed first. These data will be further process by the mathematical formula based on the parameter, and data bytes received. After acquiring sensor data, it sent to the cloud server via IDW01M1. A delay of 15 seconds implemented at the end of the function loop because ThingSpeak requires 15 seconds of interval for the data update to its cloud server.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

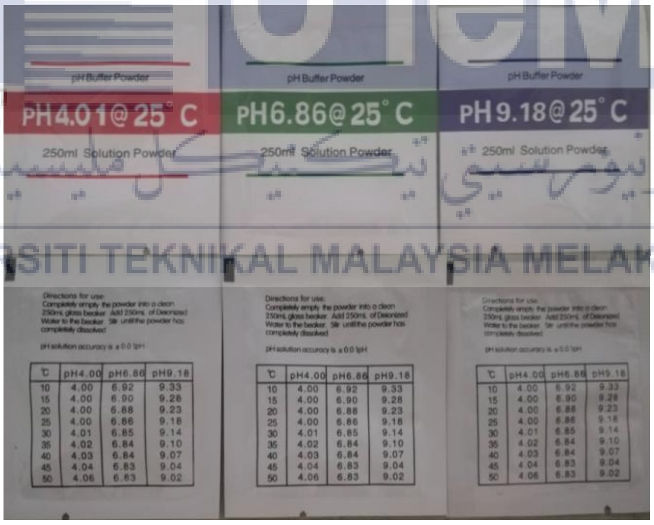
```
#include "mbed.h"
#include "SpwfInterface.h"
#include "TCPSocket.h"
#include "SEN0161.h"
#include "DS1820.h"

#define PORT 1234
#define MAX_PENDING 1
#define IP "184.106.153.149"

char* thingSpeakUrl = "http://api.thingspeak.com/update";
char* thingSpeakKey = "Z9HN57T63DBWIW2D";

SEN0161 sensor(A0);
DS1820  ds1820(A1);         // substitute D8 with the actual pin name connected to the 1-wire
bus
DigitalOut myled(LED1);
int cnt=0;

Serial pc(USBTX, USBRX);


volatile long pH=0;
volatile long voltage=0;
volatile long temp=0;
volatile long volt=0;

SpwfSAInterface spwf(D8, D2, false);


int http_demo(long pH, long temp)
{
    TCPSocket socket(&spwf);
    char buffer[512];
    char message[80];
    int err;
    printf("Sending HTTP Data to thingspeak...\r\n");

    // Open a socket on the network interface, and create a TCP connection to thingspeak
    //socket.open(&spwf);
    err=socket.connect(IP,80); // connecting to thingspeak
    if(err!=0) {
        pc.printf("\r\nCould not connect to Socket, err = %d!!\r\n", err);
        return -1;
        }
    else
        pc.printf("\r\nconnected to host server\r\n");

        // Send a simple http request
        sprintf(message,"field1=%d&field2=%d,pH,temp);

    printf("Message Length=%d\r\n",(int)strlen(message));
    sprintf(buffer,"POST    /update    HTTP/1.1\r\nHost:    api.thingspeak.com\r\nConnection:
close\r\nX-THINGSPEAKAPIKEY:          %s\r\nContent-Type:          application/x-www-form-
urlencoded\r\nContent-Length: %d\r\n\r\n%s",thingSpeakKey,(int)strlen(message),message);
    pc.printf("Request to %s\r\n", buffer);
    //char sbuffer[] = "GET / HTTP/1.1\r\nHost: www.arm.com\r\n\r\n";
    int scount = socket.send(buffer, (int)strlen(buffer));
    printf("sent %d [%.*s]\r\n", scount, strstr(buffer, "\r\n")-buffer, buffer);

    // Recieve a simple http response and print out the response line
    char rbuffer[64];
    int rcount = socket.recv(rbuffer, sizeof rbuffer);
    printf("recv %d [%.*s]\r\n", rcount, strstr(rbuffer, "\r\n")-rbuffer, rbuffer);

    // Close the socket to return its memory and bring down the network interface
    socket.close();
    return 0;
}

int main()
{
    int err;
    char * ssid = "terry1";
    char * seckey = "123456789";
    pc.printf("\r\nNucleo F411RE + WiFi + Thingspeak\r\n");
    pc.printf("\r\nconnecting to AP\r\n");

    if(spwf.connect(ssid, seckey, NSAPI_SECURITY_WPA2)) {
        pc.printf("\r\nnow connected\r\n");
    } else {
        pc.printf("\r\nerror connecting to AP.\r\n");
        return -1;
    }

    const char *ip = spwf.get ip address();
    const char *mac = spwf.get_mac_address();

    pc.printf("\r\nIP Address is: %s\r\n", (ip) ? ip : "No IP");
    pc.printf("\r\nMAC Address is: %s\r\n", (mac) ? mac : "No MAC");

    pc.printf("\r\n--Starting--\r\n");

    if(ds1820.begin())
    {
        while(1)
        {
            wait(1.0);
            myled = !myled;

                pc.printf("----------------\r\n");
                float voltage=sensor.get_pH();
                float pH=-6*voltage+16;
                printf("\n\r PH is %f\n", pH);
                //float voltage= (sensor.get_pH())/3.5f;
                //printf("\n\r mV is %f\n", voltage);
                ds1820.startConversion();   // start temperature conversion
                float temp=ds1820.read();
                printf("\n\r temp = %3.1f\r\n", temp);     // read temperature            =
                int err=http_demo(pH,voltage,temp,volt);

                    if(err==0)
                        pc.printf("Thingspeak update completed successfully\r\n");
                    else
                        pc.printf("Error occurred %d\r\n",err);
        }
    }
        else
        printf("No DS1820 sensor found!\r\n");
}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

### 3.1.3.3 Implementation of Water Monitoring System (Offline)

After the implementation, hardware and software are done. The prototype is tested by measure three different water samples solution with varying values of ph (acid, neutral, and alkaline). The figure below shows the pH buffer powder used in this project as a benchmark or standard reference for the parameter's pH values. Throughout this buffer powder, three different pH water samples solution is made—the instructions and the details of the pH value that corresponds to the temperature values are stated too. The microcontrollers measured the parameter pH values and computed the result to determine the exact pH value from the water samples solution.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



Directions for use:
Completely empty the powder into a clean 250mL glass beaker. Add 250mL of Deionized Water to the beaker. Stir until the powder has completely dissolved.

pH solution accuracy is ± 0.0 1pH

| ℃ | pH4.00 | pH6.86 | pH9.18 |
|---|---|---|---|
| 10 | 4.00 | 6.92 | 9.33 |
| 15 | 4.00 | 6.90 | 9.28 |
| 20 | 4.00 | 6.88 | 9.23 |
| 25 | 4.00 | 6.86 | 9.18 |
| 30 | 4.01 | 6.85 | 9.14 |
| 35 | 4.02 | 6.84 | 9.10 |
| 40 | 4.03 | 6.84 | 9.07 |
| 45 | 4.04 | 6.83 | 9.04 |
| 50 | 4.06 | 6.83 | 9.02 |

**Figure 3.7 Three different pH Buffer Powder.**

Besides, to further make the benchmark to be more accurate, the ethanol solution is used in this project as shown in the figure below. It acts as the litmus paper, where four drops of ethanol solution will drop into each of these pH buffer solutions, and the pH of the water samples solution is determined according to the color changes. It does prove the pH of the solution.

Unfortunately, this only gives a rough guess towards the pH value. It is not strong enough to convince the pH value that the microcontroller obtained either is correct or not.

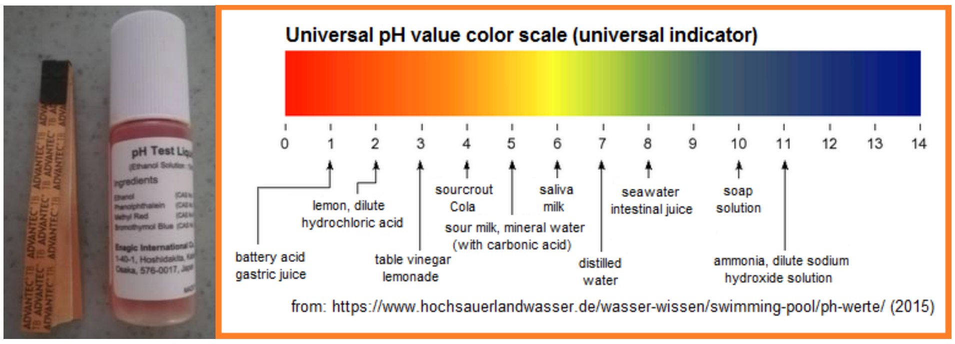


**Figure 3.8 Ethanol Solution and Litmus Paper.**

Hence, another device has been purchased and used, which is the analytical instruments that can measure the pH, conductivity (EC), total dissolved solids (TDS), and temperature. With this instrument, the problem of getting a reference pH benchmark parameter reading is solved. Figures below show the water samples solution mixed with the ethanol solution and tested with the analytic device to ensure and set all the reference parameters reading before the solutions are tested with the water monitoring system.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

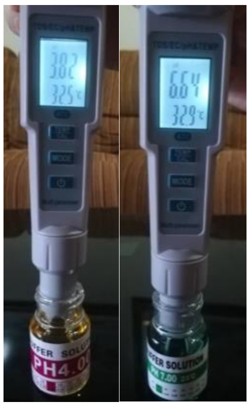


**Figure 3.9 pH Buffer solution tested with ethanol solution and analytical instruments.**

## 3.2 Analysis of Time Response

A second-order system exhibits a wide range of responses that must be analyzed and described. Varying a first-order system's parameter simply changes the speed of the reaction; changes in the parameters of a second-order system can change the form of the response. For example, a second-order system can display characteristics much like a first-order system or, depending on component values, display damped or pure oscillations for its transient response. The term in the numerator is simply a scale or input-multiplying factor that can take on any amount without affecting the form of derived results.

By assigning appropriate values to parameters a and b, we can show all possible second-order transient responses. The unit step response than can be found using $C(s) = R(s)\,G(s)$, where $R(s) = \frac{1}{s}$, followed by a *partial-fraction expansion* and the *inverse Laplace* transform.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Overdamped responses:

Poles: Two real at $-\sigma_1, \sigma_2$.

Natural response: Two exponentials with time constants equal to the reciprocal of the pole locations, or

$$c_n(t) = K_1 e^{-\sigma 1 t} + K_2 e^{-\sigma 2 t} \qquad 3.1$$

Underdamped responses:

Poles: Two complexes at $-\sigma_d, j\omega_d$.

Natural response: Damped sinusoid with an exponential envelope whose time constant is equal to the reciprocal of the pole's real part. The radian frequency of sinusoid, the damped frequency of oscillation, is similar to the imaginary part of the poles, or

$$c_n(t) = Ae^{-\sigma dt} cos\ \omega_d t - \phi \qquad 3.2$$

Undamped responses:

Poles: Two imaginary at $\pm j\ \omega_1$

Natural response: Undamped sinusoid with radian frequency equal to the imaginary part of the poles, or

$$c_n(t) = A\ cos\ (\omega_1 t - \phi) \qquad 3.3$$

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Evaluation of $T_p$

$$T_p = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} \qquad 3.4$$

Evaluation of $\%OS$

$$\%OS = \frac{C_{max} - C_{final}}{C_{final}} \times 100 = e^{-\left(\frac{\zeta\pi}{\sqrt{1-\zeta^2}}\right)} \times 100 \qquad 3.4$$

*where*

$$C_{max} = 1 + e^{-\left(\frac{\zeta\pi}{\sqrt{1-\zeta^2}}\right)}$$

$and\ for\ the\ unit\ step\ used;$

$$C_{final} = 1$$

$the\ inverse\ of\ this\ equation\ allows\ to\ solve\ for\ \zeta\ by:$

$$\zeta = \frac{-\ln(\%OS/100)}{\sqrt{\pi^2 + [\ln\left(\%\frac{OS}{100}\right)]^2}}$$

<u>Evaluation of $T_s$</u>

$For\ criteria\ between\ \pm 2\%,$

$$T_s = \frac{4}{\zeta\omega_n} \qquad 3.5$$

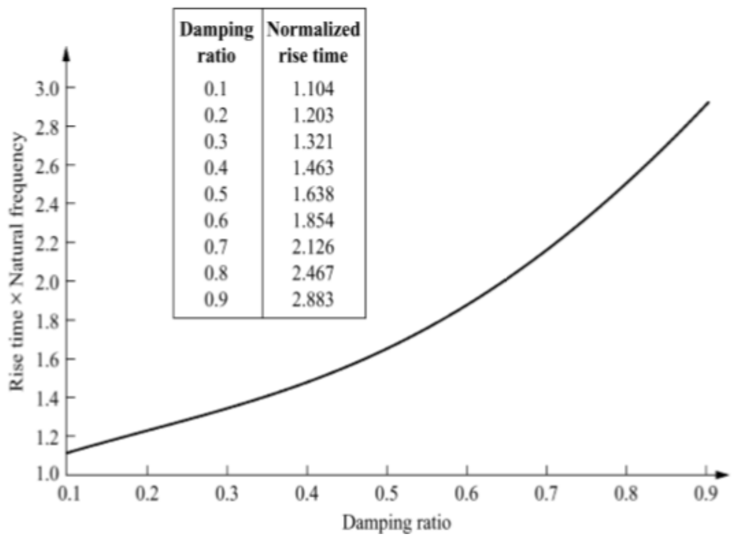
$For\ criteria\ between\ \pm 5\%,$

$$T_s = \frac{4}{\zeta\omega_n} \qquad 3.6$$

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

<u>Evaluation of $T_r$</u>

A precise analytical relationship between rising time and damping ratio, ζ cannot be found. However, we can see the rise time using a computer. Figure 3.6 illustrates the resulting curve obtained relating the variation of $w_n t_r$ and the damping ratio, ζ



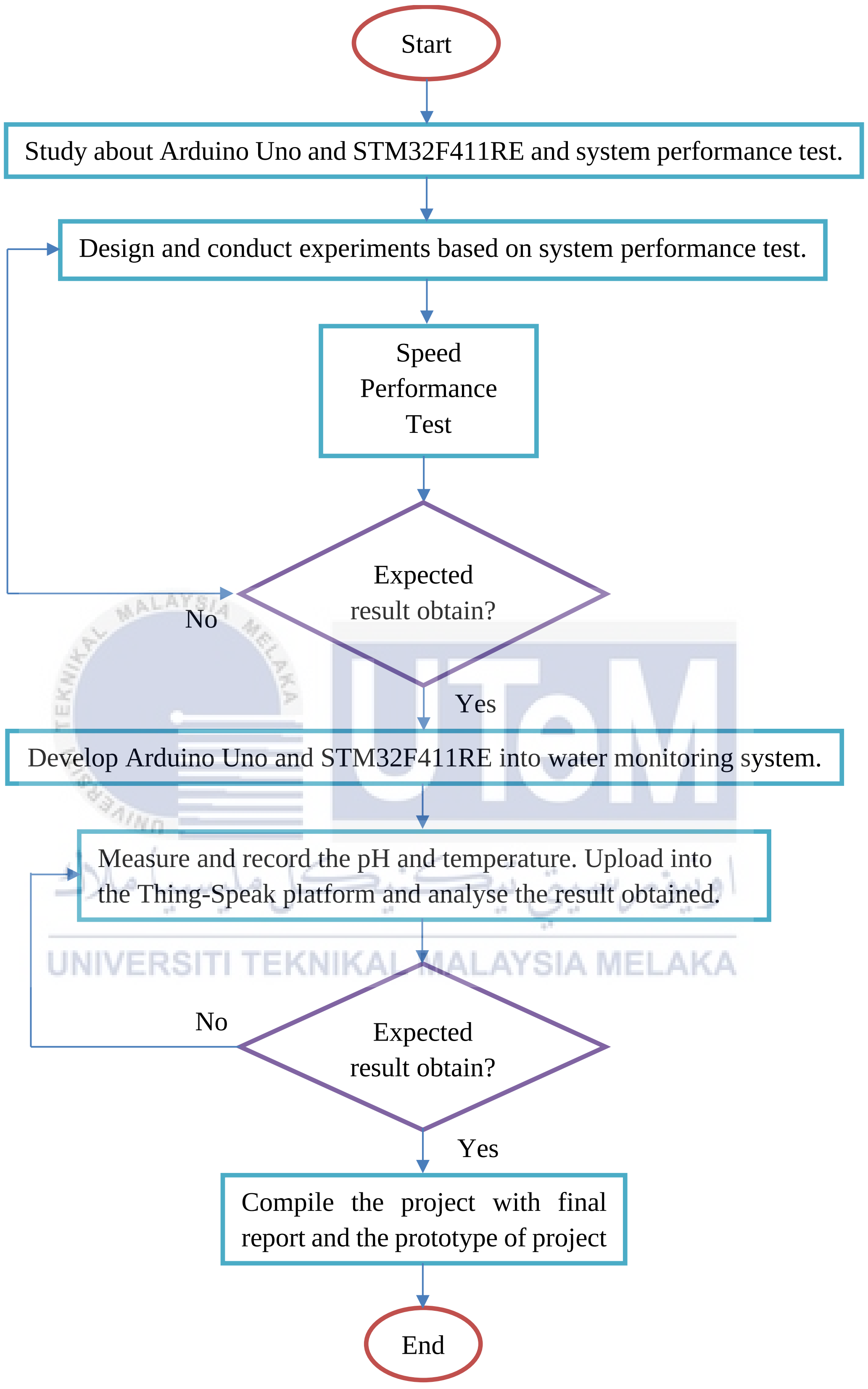
| Damping ratio | Normalized rise time |
|---|---|
| 0.1 | 1.104 |
| 0.2 | 1.203 |
| 0.3 | 1.321 |
| 0.4 | 1.463 |
| 0.5 | 1.638 |
| 0.6 | 1.854 |
| 0.7 | 2.126 |
| 0.8 | 2.467 |
| 0.9 | 2.883 |

**Figure 3.10 Illustration of the variation of $w_n t_r$ and the damping ratio, ζ.**

## 3.3 Description of Project Implementation

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

To obtain the result for further analysis, project testing based on the developed project prototype produced. Besides, the functionality of the system tested to ensure that the expected result coherence with the project objective achieved. Figure 3.3.1 and Table 3.3.1 below show the flowchart and Gantt chart of the overall project implementation, respectively.


Start
Study about Arduino Uno and STM32F411RE and system performance test.
Design and conduct experiments based on system performance test.
Speed Performance Test
Expected result obtain?
No
Yes
Develop Arduino Uno and STM32F411RE into water monitoring system.
Measure and record the pH and temperature. Upload into the Thing-Speak platform and analyse the result obtained.
Expected result obtain?
No
Yes
Compile the project with final report and the prototype of project
End

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Figure 3.11 Flow chart of the project.**

# CHAPTER 4

# RESULTS AND DISCUSSION

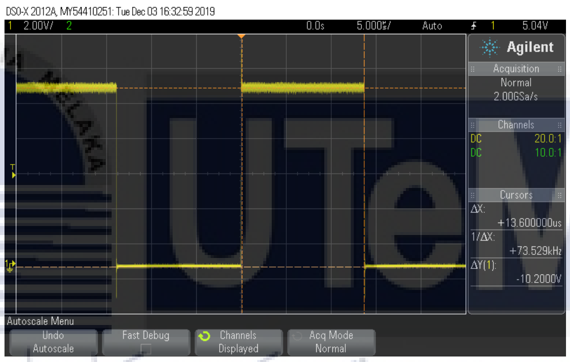
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

This chapter will discuss the outcome of this project from the previous methodology. The product of hardware and software implementation on both microcontrollers, Arduino UNO and STM32F411RE with water monitoring system, will be displayed in the form of a picture. The collected data will be shown in a graphical and tabulated form so that it is user friendly. The findings of this section used to review the objective of this project.

## 4.1 Minimum latency comparison

To execute minimum latency measurement, one of the pins from the microcontroller chose as input and another as an output. A wire linked both pins, and between the connection, the oscilloscope attached to this connection. Attached to the input pin was an interrupt handler that reacts on each edge. It intends to set an output pin to input pin's inverse value. It leads to an infinite sequence of value changes as the output is connected back to the origin, thereby producing a square wave with a period that is double the typical latency.

After implementing the code that wrote in the previous chapter, the waveform and the data obtained. Figure 4.4, Figure 4.5, and Figure 4.6 below show the waveform, which generated as the changes of the signal of the output pin towards the input pin as the interrupt handler as it attached to the input pin.


DSO-X 2012A, MY54410251: Tue Dec 03 16:32:59 2019
1 2.00V/ 2 0.0s 5.000µs/ Auto 1 5.04V
Acquisition Normal 2.00GSa/s
Channels DC 20.0:1 DC 10.0:1
Cursors ΔX: +13.600000us 1/ΔX: +73.529kHz ΔY(1): -10.2000V

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Figure 4.1 Latency test with pure wiring (Arduino UNO)**

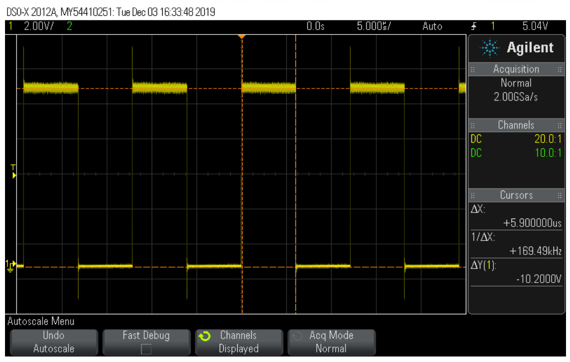

DSO-X 2012A, MY54410251: Tue Dec 03 16:33:48 2019
1 2.00V/ 2 0.0s 5.000µs/ Auto 1 5.04V
Acquisition Normal 2.00GSa/s
Channels DC 20.0:1 DC 10.0:1
Cursors ΔX: +5.900000us 1/ΔX: +169.49kHz ΔY(1): -10.2000V

**Figure 4.2: Latency test with direct manipulation of the port register (Arduino UNO)**

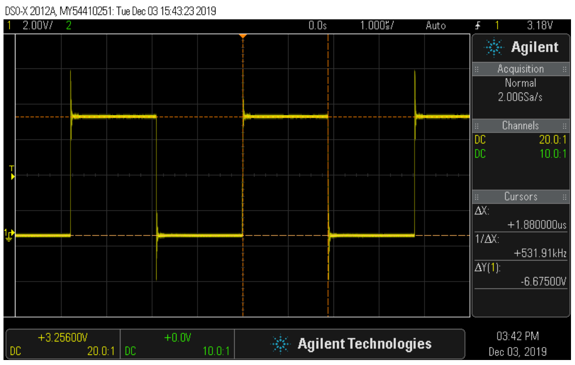


**Figure 4.3: Latency test (STM32F411RE)**

From the waveform generated by the oscilloscope, the time taken for the microcontrollers to respond or to change towards the input interrupt handler's signal is much clearer to be seen. Table 4.1 below shows an overview of the comparative latency result.

**Table 4.1: Overview of minimum latency**

| Types of microcontroller | Level of Coding | Latency |
|---|---|---|
| Arduino UNO | Advance | $5.9\mu s$ |
| STM32F411RE | Advance | $1.88\mu s$ |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

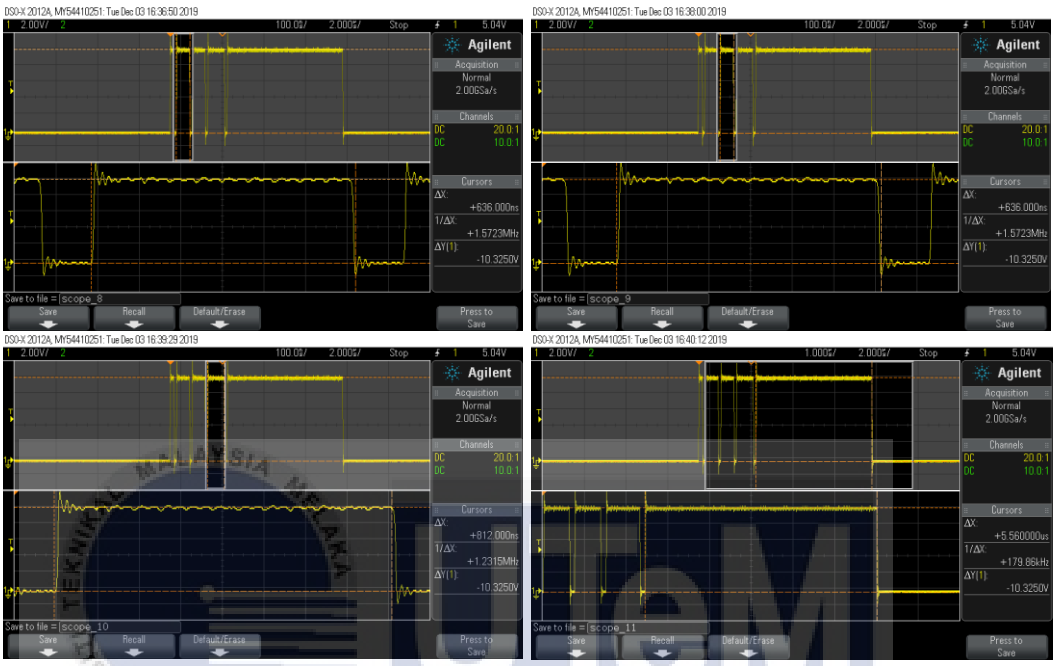
From these data, it can be clearly state that the Arduino UNO latency is much higher than the STM32F411RE. The controller STM32F411RE latency is $1.88\mu s$. While for the Arduino UNO, even though it has two types of level coding, it still having a higher latency rate than STM32F411RE, which is $5.9\mu s$ (advance). Arduino UNO has three times higher latency than STM32F411RE. The higher the latency, the longer the time taken/ interval between the stimulation and response. It also indicates that the Arduino UNO is slower than the STM32F411RE. It is because the STM32F411RE only requires only 1.88$\mu s$ to react towards each of the interrupt signals. While for the Arduino UNO R3, it requires 5.9$\mu s$ to respond to each of the interrupt signals. Besides, this also means that the STM32F411RE has a higher response rate as compared to Arduino UNO R3 that can fast enough to react towards a signal given and response to an event. As to strengthen the results obtained, the data compared with researched the data. Therefore, from this result, it has answered/ proved that the STM32F411RE is faster than the Arduino UNO in the latency test.

## 4.2 Integer arithmetic comparison

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

I programmed one pin as an output pin to measure the execution time of the mathematical procedures, and raised the output before the operation and then lowered it. The pulse produced by each phase and the length of the pulse is equivalent to the math procedure execution period plus the time needed to increase the output and decrease it. In the first pulse, the calculation process missed to compensate for the period to move, and the score reduced. The first pulse thus represents the time that must be subtracted from the other pulses to have a pure length of the math operation. Addition, subtraction, multiplication, and division encapsulated in the corresponding pulses.

The coding for the integer math operation code, which wrote in the previous chapter, uploaded into both microcontrollers, Arduino UNO and STM32F411RE. Figure 4.4, Figure 4.5, Figure 4.6, waveforms generated.



UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Figure 4.4: Arduino UNO R3-Timing test 8bit

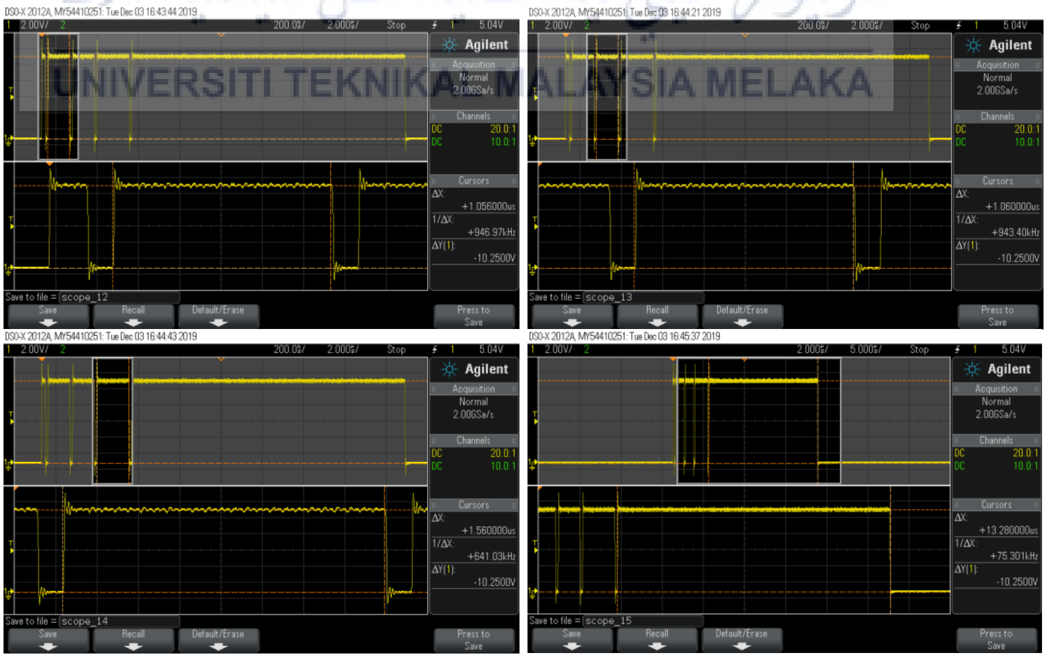


Figure 4.5: Arduino UNO R3-Timing test 16 bit

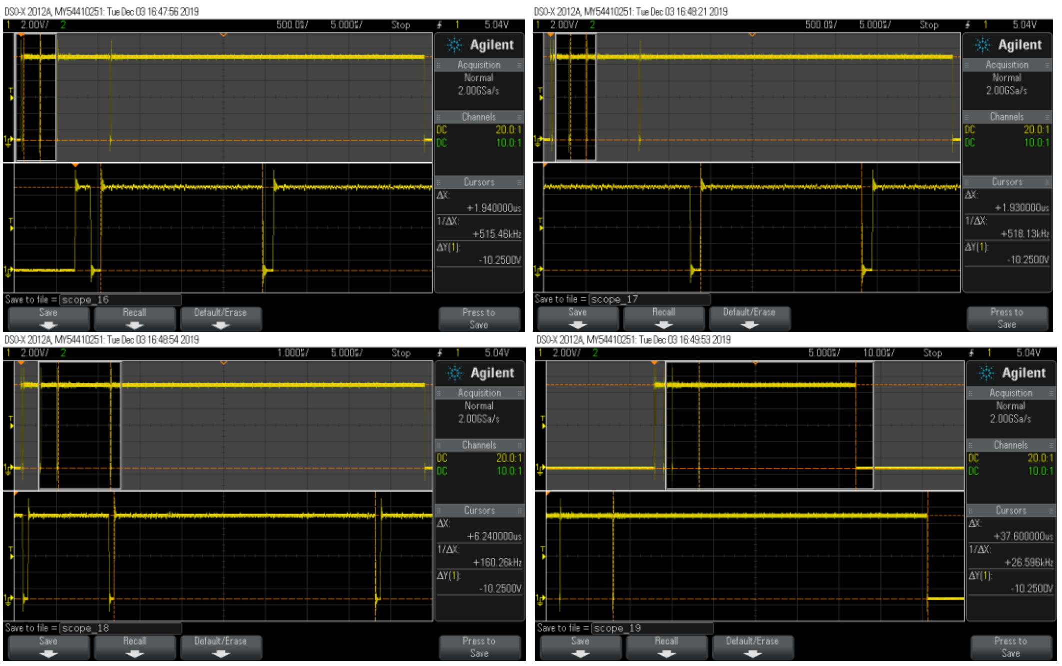


**Figure 4.6: Arduino UNO R3-Timing test 32 bit**

UTeM UNIVERSITI TEKNIKAL MALAYSIA MELAKA

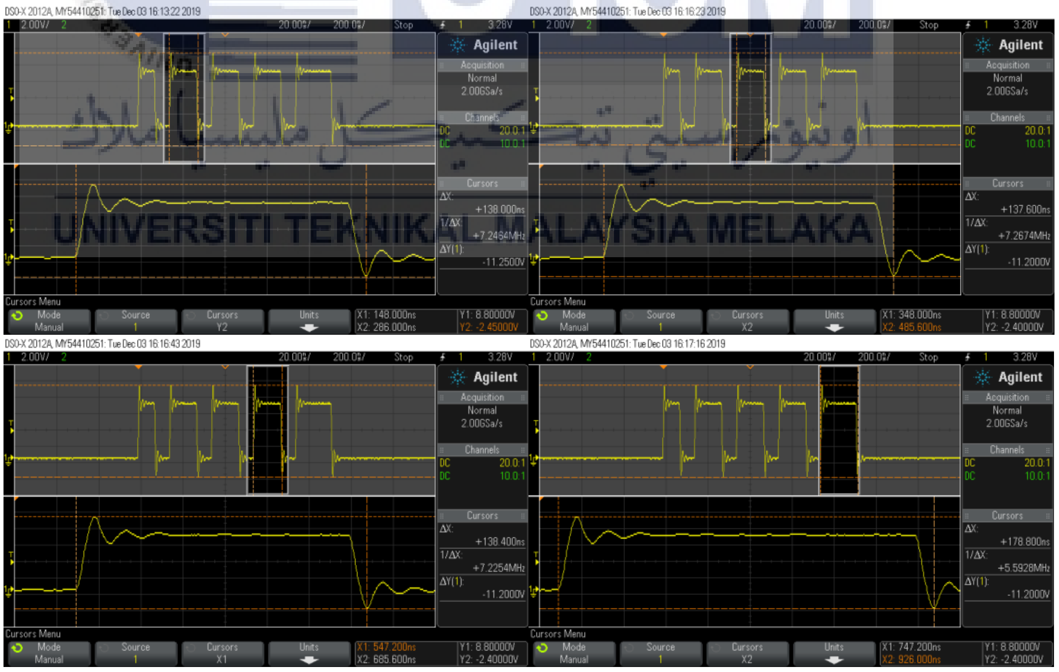


**Figure 4.7: STM32F411RE-Timing test 32 bit**

From these generated waveforms, all of the time taken for each integer math operation tabulated in table 4.2 below.

**Table 4.2: Overview of integer math operation**

| Types of Microcontrollers | Data Bit Width | Types Arithmetic Operation | Time Taken for Execution |
|---|---|---|---|
| Arduino UNO R3 | 8 | Additional | $636ns$ |
| | | Subtraction | $636ns$ |
| | | Multiplication | $812ns$ |
| | | Division | $5.56\mu s$ |
| | 16 | Additional | $1.056\mu s$ |
| | | Subtraction | $1.06\mu s$ |
| | | Multiplication | $1.56\mu s$ |
| | | Division | $13.28\mu s$ |
| | 32 | Additional | $1.94\mu s$ |
| | | Subtraction | $1.93\mu s$ |
| | | Multiplication | $6.24\mu s$ |
| | | Division | $37.6\mu s$ |
| STM32F411RE | 32 | Additional | $138ns$ |
| | | Subtraction | $137.6ns$ |
| | | Multiplication | $138.4ns$ |
| | | Division | $178ns$ |

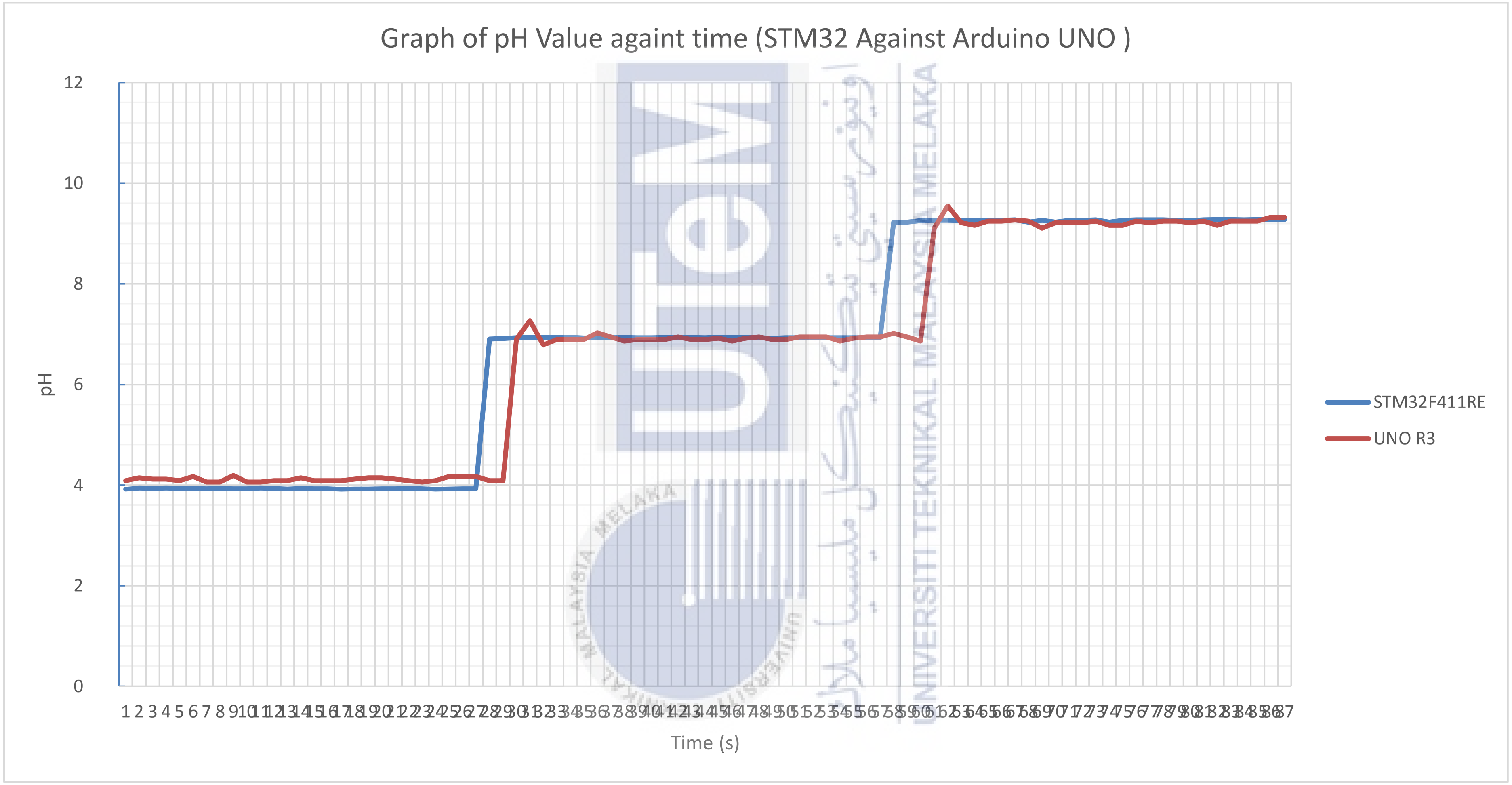
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

From these results, the STM32F411RE has a significant high performance than Arduino UNO R3. This is because STM32F411RE in 32-bit integer math operations additional is $138\text{n}s$, Subtraction $137.6\text{n}s$, Multiplication $138.4\text{n}s$ and Division is $178\text{n}\mu s$. While for the Arduino UNO with 32 bits, the integer operation for additional is $1.94\mu s$, Subtraction $1.93\mu s$, Multiplication $6.24\mu s$ and Division is $37.6\mu s$. It indicates that the STM32F411RE microcontroller has a faster arithmetic calculation process than the Arduino UNO R3, which faster enough to do the arithmetic math operation in a short time (Additional, Subtraction, Multiplication, and Division). The system performance of STM32F411RE is much faster than the Arduino UNO R3. The result obtained is then compared with the result published in the research paper. Hence, the result got once again proved that the STM32F411RE is faster than Arduino UNO in integer math operation.

## 4.3 Wireless performance comparison

By uploading the code in the previous chapter, both microcontrollers are able to integrate with the sensors to collect the data and upload it into the ThingSpeak IoT platform. The uploaded data is then downloaded and transfer into Microsoft Office Excel for further interpretation or analysis. From the data, the pH data have the apparent potential to be analyzed while the temperature data is slightly constant and stable reading for each sample; this is due to the data collected in the same temperature condition. Hence, it analyzed the pH data collected by both microcontrollers. The time response analyzed by converting the data into a graph, as shown in Figure 4.13 below.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



UTeM
UNIVERSITI TEKNIKAL MALAYSIA MELAKA
اونيۏرسيتي تيكنيكل مليسيا ملاك

**Figure 4.8 Graph of pH Value Against Time (STM32F411RE & Arduino UNO).**

From the graph, time response analyzed by determining the peak time, percent overshoot, rise time, and settling time, etc. The system divided into two analysis which is Acid-to-Neutral and Neutral-to-Alkaline. The data calculated and tabulated, as shown in Table 4.3 and Table 4.4 below (Acid-to-Neutral).

**Table 4.3 Time response analysis of STM32F411RE (Acid-to-Neutral).**

| STM32F411RE | | | | |
|---|---|---|---|---|
| | Mean | | | |
| acid | 3.963743033 | | | |
| neutral | 6.9294178 | | | |
| alkaline | 9.258266633 | | | |
| Acid to Neutral | | | | |
| | Time | minus 27s | $C_{max}$ | $C_{final}$ |
| Peak time, $T_p$ | $31s$ | $4s$ | 6.940512 | 6.9294178 |
| Percent Overshoot, $\%OS$ | 0.160102917 | | | |
| Damping ratio, $\zeta$ | 0.898683729 | | | |
| Natural Frequency, $w_n$ | 1.790704678 | | | |
| | Refer to table | | | |
| Normalized rise time | 2.883 | | | |
| Rise Time, $T_r$ | 1.609980716 | | | 28.60998072 |
| Settling Time, $T_s$ | 2.485588006 | | | 29.48558801 |
| Characteristics of Response | | | | |
| 0<ζ<1 | Underdamped Response | | | |
| $G_1(s)$ | $\dfrac{3.2066}{s^2 + 3.2186s + 3.2066}$ | | | |
| poles | $-1.6093 \pm 0.7853i$ | | | |
| zeros | - | | | |

**Table 4.4 Time response analysis of Arduino UNO R3 (Acid-to-Neutral).**

| Arduino UNO R3 | | | | |
|---|---|---|---|---|
| | Mean | | | |
| acid | 4.110517517 | | | |
| neutral | 6.834200225 | | | |
| alkaline | 9.233890407 | | | |
| Acid to Neutral | | | | |
| | seconds | minus 29s | $C_{max}$ | $C_{final}$ |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

| | | | | |
|---|---|---|---|---|
| Peak time, $T_p$ | 31 | 4 | 7.268541 | 6.922752587 |
| Percent Overshoot, $\%OS$ | 4.994955526 | | | |
| Damping ratio, $\zeta$ | 0.690228489 | | | |
| Natural Frequency, $w_n$ | 1.085416541 | | | |
| | Refer to table | | | |
| Normalized rise time | 2.1601 | | | |
| Rise Time, $T_r$ | 1.990111555 | | | 30.99011155 |
| Settling Time, $T_s$ | 5.339132205 | | | 34.3391322 |
| Characteristics of Response | | | | |
| 0<ζ <1 | Underdamped Response | | | |
| $G_1(s)$ | $\frac{1.1781}{s^2 + 1.4984 + 1.1781}$ | | | |
| poles | $-0.7492 \pm 0.7854i$ | | | |
| zeros | - | | | |

From the tabulated result, analyzed Acid-to-Neutral of time response. The system is said to be a fast system performance where a system has a low rising time or a system that has a little settling time.

**Table 4.5 Rise time and settling time (Acid-to-Neutral)**

| Microcontroller | Rise time | Settling time |
|---|---|---|
| STM32F411RE | $1.61s$ | $2.49s$ |
| Arduino UNO R3 | $1.99s$ | $5.33s$ |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

From the table, Acid-to-Neutral time response analysis, the STM32F411RE has the rise time at $1.61s$, and the Arduino UNO R3 has the rise time at $1.99s$. From this result, the STM32F411RE has slightly higher rise time compared to the Arduino UNO R3 microcontroller for about $0.38s$. It means that the Arduino UNO R3 has a short delay of $0.38s$ in uploading the data compared to Arduino UNO R3. For the settling time, the STM32F411RE has settling time at $2.49s$, whereas the Arduino

UNO R3 has the settling time at $5.33s$. For this time, the STM32F411RE became faster to settling down the reading compared to Arduino UNO R3. It indicates that the STM32F411RE is faster than Arduino UNO R3 about $2.84s$ for the transient's damped oscillations to reach and stay within $\pm 2\%$ of the steady state value. So, for the overall, the Arduino UNO R3 requires $3.34s$ for the system to response towards the changes (waveform to go from 0.1 to 0.9 of its final value) and maintain the transient's damped oscillation at $\pm 2\%$ of the steady state value. While for the STM32F411RE, it requires $0.88s$. Which means that STM32F411RE system performance is faster to compute towards the input signal and produce the output data than Arduino UNO R3. As it requires only a short time to respond to any changes at the parameters and maintain the reading rapidly, which only requires $0.88s$.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

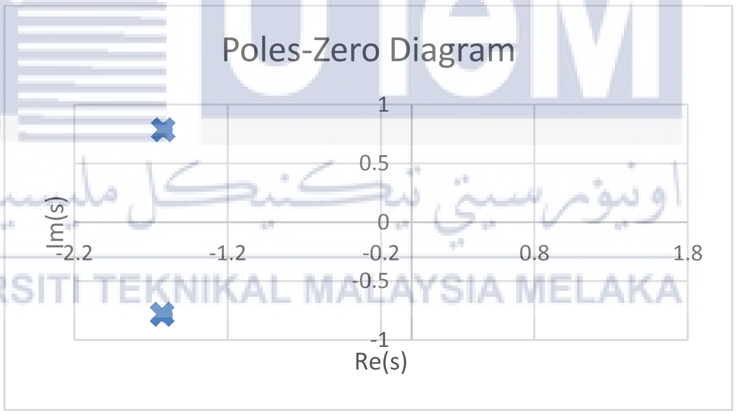


**Figure 4.9 Pole-zero diagram of STM32F411RE (Acid-to-Neutral).**

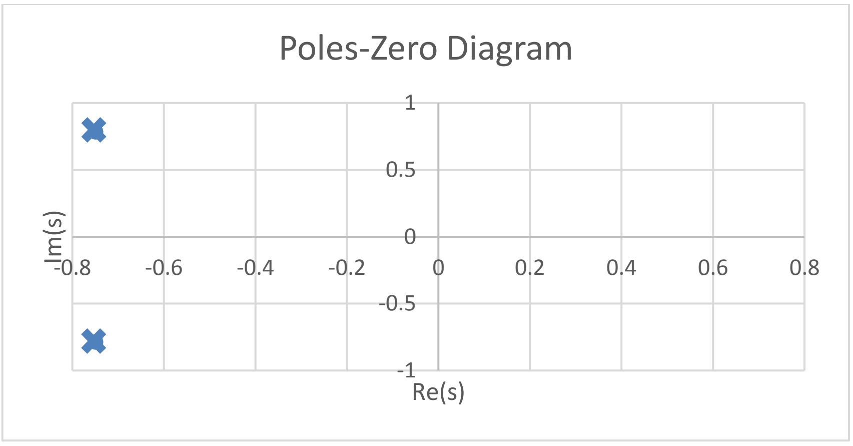


**Figure 4.10 Pole-zero diagram of Arduino UNO R3 (Acid-to-Neutral).**

Besides, from the analysis of time response, the transfer function of both systems is formed. From the transfer function, the poles and zeros were able to be determined to plot the pole-zero diagram, as shown in Figure 4.14 and Figure 4.15. From these pole-zero diagrams, both of the systems do not have any zero, but both methods have two poles, and they are in the left half-plane. Hence, the system is said to be a stable closed-loop system.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Table 4.6 Percent Overshoot (Acid-to-Neutral)**

| Microcontroller | Overshoot Percent |
|---|---|
| STM32F411RE | 0.18% |
| Arduino UNO R3 | 1.91% |

Furthermore, the STM32F411RE has an overshoot percent of 0.18%, while the Arduino UMO R3 has an overshoot percent of 1.91%. Hence, the STM32F411RE is said to be more stable, static, and consistent compared to Arduino UNO R3. From these analysis results, the STM32F411RE is slightly faster and durable than the Arduino UNO R3 for the time response type of Acid-to-Neutral.

**Table 4.7 Damping ratio (Acid-to-Neutral)**

| Microcontroller | Damping ratio, ζ |
|---|---|
| STM32F411RE | 0.4166 |
| Arduino UNO R3 | 0.7832 |

Finally, both systems have the same characteristics of response, which are Underdamped Response. It is due to their damping ratio, ζ, which is more than 0 but less than 1 (0<ζ <1).

In the meantime, the analysis time response of Neutral-to-Alkaline has also analyzed. Tables 4.5 and 4.6 below show the tabulation of the calculation result.

**Table 4.8 Time response analysis of STM32F411RE (Neutral-to-Alkaline).**

| STM32F411RE | | | | |
|---|---|---|---|---|
| | Mean | | | |
| acid | 3.963743033 | | | |
| neutral | 6.9294178 | | | |
| alkaline | 9.258266633 | | | |
| Neutral to Alkaline | | | | |
| | Time | minus 57s | $C_{max}$ | $C_{final}$ |
| Peak time, $T_p$ | 67 | 10 | 9.275671 | 9.258266633 |
| Percent Overshoot, $\%OS$ | 0.187987317 | | | |
| Damping ratio, $\zeta$ | 0.894238069 | | | |
| Natural Frequency, $w_n$ | 0.701888143 | | | |
| | Refer to table | | | |
| Normalized rise time | 2.883 | | | |
| Rise Time, $T_r$ | 4.107492099 | | | 61.1074921 |
| Settling Time, $T_s$ | 6.372926818 | | | 63.37292682 |
| Characteristics of Response | | | | |
| 0<ζ <1 | Underdamped Response | | | |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

| | |
|---|---|
| $G_2(s)$ | $\frac{0.4926}{s^2+1.2553s+0.4926}$ |
| poles | $-0.6277 \pm 0.3141i$ |
| zeros | - |

**Table 4.9 Time response analysis of Arduino UNO R3 (Neutral-to-Alkaline).**

| Arduino UNO R3 | | | | |
|---|---|---|---|---|
| | Mean | | | |
| acid | 4.110517517 | | | |
| neutral | 6.834200225 | | | |
| alkaline | 9.233890407 | | | |
| Neutral to Alkaline | | | | |
| | Time | minus 60s | $C_{max}$ | $C_{final}$ |
| Peak time, $T_p$ | 67s | 7s | 9.546515 | 9.233890407 |
| Percent Overshoot, $\%OS$ | 3.385621648% | | | |
| Damping ratio, $\zeta$ | 0.733031939 | | | |
| Natural Frequency, $w_n$ | 0.659810004 | | | |
| | Refer to table | | | |
| Normalized rise time | 2.3988 | | | |
| Rise Time, $T_r$ | 3.635592042 | | | 60.63559204 |
| Settling Time, $T_s$ | 8.270241608 | | | 65.27024161 |
| Characteristics of Response | | | | |
| 0<ζ<1 | Underdamped Response | | | |
| $G_2(s)$ | $\frac{0.4353}{s^2 + 0.9673s + 0.4353}$ | | | |
| poles | $-0.4837 \pm 0.4488i$ | | | |
| zeros | - | | | |

**Table 4.10 Rise time and settling time (Neutral-to-Alkaline)**

| Microcontroller | Rise time | Settling time |
|---|---|---|
| STM32F411RE | $4.11s$ | $6.37s$ |
| Arduino UNO R3 | $3.64s$ | $8.27s$ |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

From the table, Neutral-to-Alkaline time response analysis, the STM32F411RE has the rise time at $4.11s$, and the Arduino UNO R3 has the rise time at $3.64s$. From this result, the STM32F411RE has slightly higher rise time compared to the Arduino UNO R3 microcontroller for about $0.47s$. It means that the STM32F411RE has a short delay of $0.47s$ in uploading the data compared to Arduino UNO R3. For the settling time, the STM32F411RE has settling time at $6.37s$, whereas the Arduino UNO R3 has the settling time at $8.27s$. For this time, the STM32F411RE became faster to settling down the reading compared to Arduino UNO R3. It indicates that the STM32F411RE is faster than Arduino UNO R3 about $1.9s$ for the transient's damped oscillations to reach and stay within $\pm 2\%$ of the steady state value. So, for the overall, the Arduino UNO R3 requires $4.63s$ for the system to response towards the changes (waveform to go from 0.1 to 0.9 of its final value) and maintain the transient's damped oscillation at $\pm 2\%$ of the steady state value. While for the STM32F411RE, it requires $2.26s$. Which means that STM32F411RE system performance is faster to compute towards the input signal and produce the output data than Arduino UNO R3. As it requires only a short time to respond to any changes at the parameters and maintain the reading rapidly, which only requires $2.26s$. Although the STM32F411RE have a short delay in responding towards the input signal at this time, the time taken for the STM32F411RE to compute and produce the output signal is way faster than the Arduino UNO R3. Once again, the STM32F411RE has a low settling time compared to the Arduino UNO R3. Hence, the STM32F411RE system is faster than the Arduino UNO R3.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

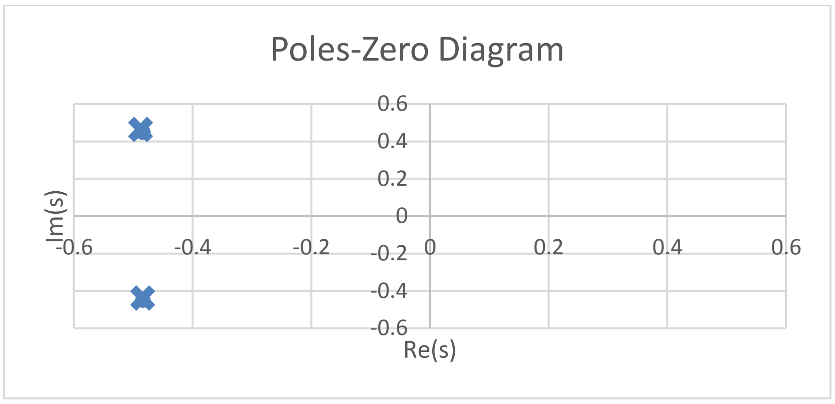


**Figure 4.11 Pole-zero diagram of Arduino UNO R3 (Neutral-to-Alkaline).**

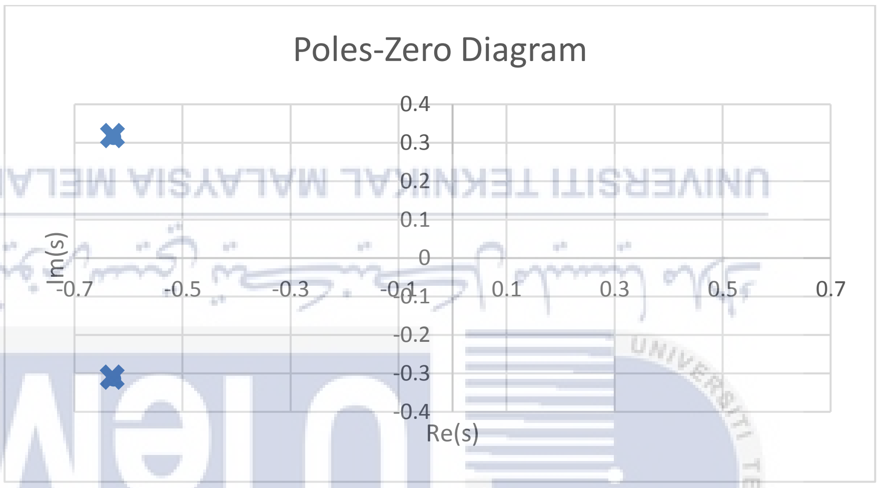


**Figure 4.12 Pole-zero diagram of STM32F411RE (Neutral-to-Alkaline).**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Besides, from the analysis of time response, the transfer function of both systems are formed. From the transfer function, the poles and zeros were able to be determined to plot the pole-zero diagram, as shown in Figure 4.16 and Figure 4.17. From these pole-zero diagrams, both of the systems do not have any zero, but both methods have two poles, and they are in the left half-plane. Hence, the system is said to be a stable closed-loop system.

**Table 4.11 Percent Overshoot (Neutral-to-Alkaline)**

| Microcontroller | Overshoot Percent |
|---|---|
| STM32F411RE | 0.19% |
| Arduino UNO R3 | 3.39% |

Furthermore, due to the overshoot percent that Arduino UNO R3 (3.39%) has that slightly higher than the STM32F411RE ( 0.19% ). It shows that the STM32F411RE is more stable, static, and consistent value compared to the Arduino UNO R3.

**Table 4.12 Damping ratio (Acid-to-Neutral)**

| Microcontroller | Damping ratio, ζ |
|---|---|
| STM32F411RE | 0.8942 |
| Arduino UNO R3 | 0.7330 |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Finally, both systems have the same characteristics of response, which are Underdamped Response. It is due to their damping ratio, ζ, which is more than 0 but less than 1 ($0<\zeta<1$).

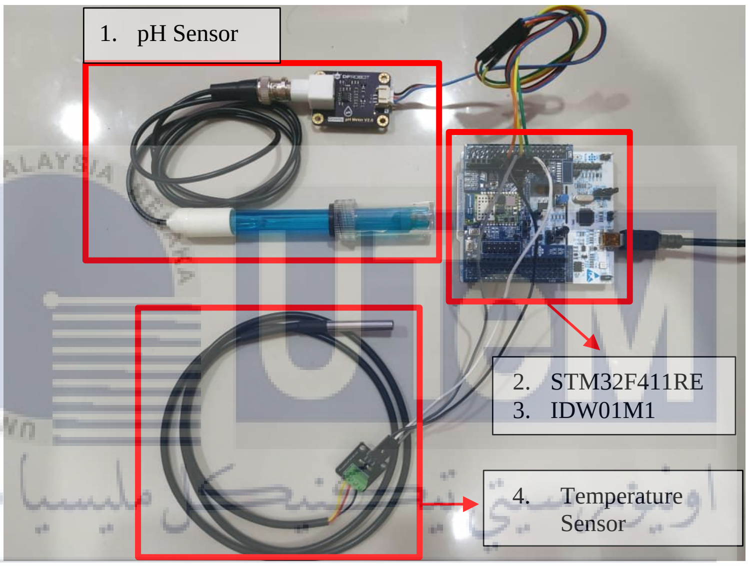
From the analysis time response of Acid-to-Neutral and Neutral-to-Alkaline, the STM32F411RE is faster and more stable than the Arduino UNO R3. From the result, it also means that the STM32F411RE has a high-performance system than Arduino UNO R3. From this conclusion, it proved and answered my objective of this project.

## 4.4 Development of Water Monitoring System

### 4.4.1 Hardware implementation of the Water Monitoring system.

A project prototype of a water monitoring system is made by connecting the same components, which are pH sensor and temperature sensor on two different microcontrollers. The hardware components used in this project along with the overview of connection. The function of each hardware component described.

#### 4.4.1.1 STM32F411RE



UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Figure 4.13 Hardware components of the Water monitoring system on the STM32F411RE microcontroller.**

1. pH Sensor (SKU: SEN0161) that enable and help data request and collect from pH probe via BNC connector to signal conversion board to measure and obtain the pH level of solutions.
2. STM32F411RE board as a microcontroller to control and perform a task with the help of other components such as data extraction, data collection, data conversion, data are storing in the cloud, etc.

3. IDW01M1 board as the Wi-Fi module board that provides internet connectivity of this project for data transmitter and receiver in the cloud.
4. Temperature sensor (DS18B20) that enable and help data request and collect to measure and obtain the temperature value of solutions.

#### 4.4.1.2 Arduino UNO R3

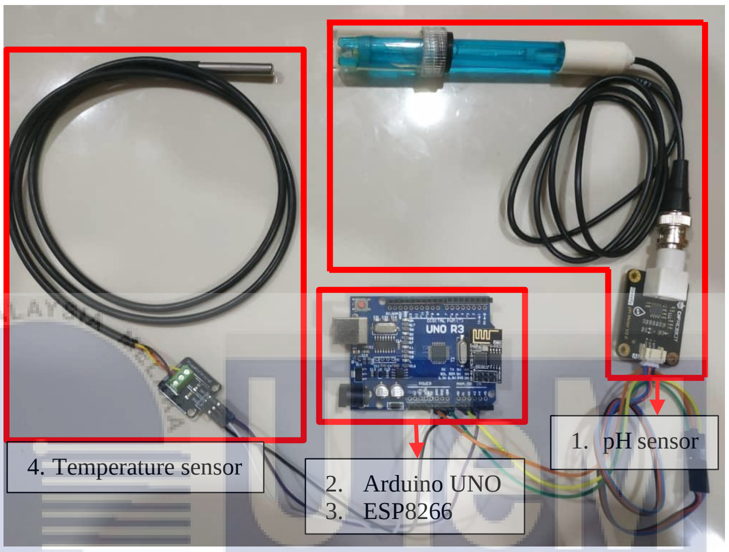


**Figure 4.14 Hardware components of the Water monitoring system on Arduino UNO R3 microcontroller.**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

1. pH Sensor (SKU: SEN0161) that enable and help data request and collect from pH probe via BNC connector to signal conversion board to measure and obtain the pH level of solutions.
2. Arduino UNO R3 board as a microcontroller to control and perform a task with the help of other components such as data extraction, data collection, data conversion, data are storing in the cloud, etc.
3. ESP8266 board as the Wi-Fi module board that provides internet connectivity of this project for data transmitter and receiver in the cloud.

4. Temperature sensor (DS18B20) that enable and help data request and collect to measure and obtain the temperature value of solutions.

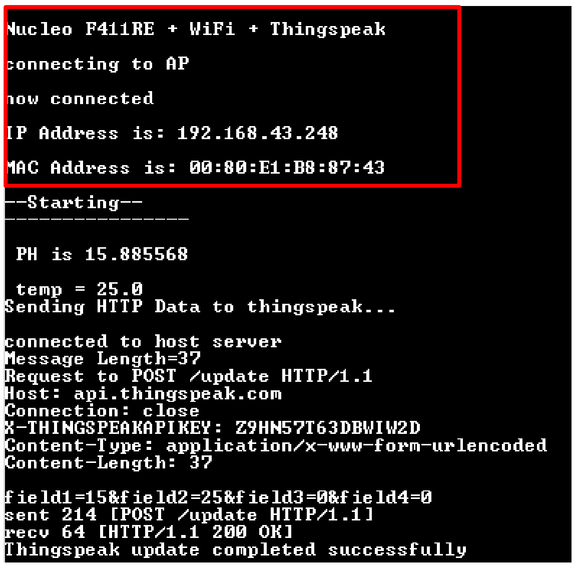
### 4.4.2 Software implementation of the Water Monitoring System.

The Arduino UNO R3 and STM32F411RE coding that implemented in the previous chapter uploaded to the Arduino board and STM32F411RE using Arduino IDE software and Mbed compiler, respectively. The Serial monitor in Arduino IDE and Tera Term used for inspecting or observe the traffic of serial port in Arduino board and STM32 board respectively to ensure the coding is working in an expected sequence,

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 4.4.2.1 STM32F411RE

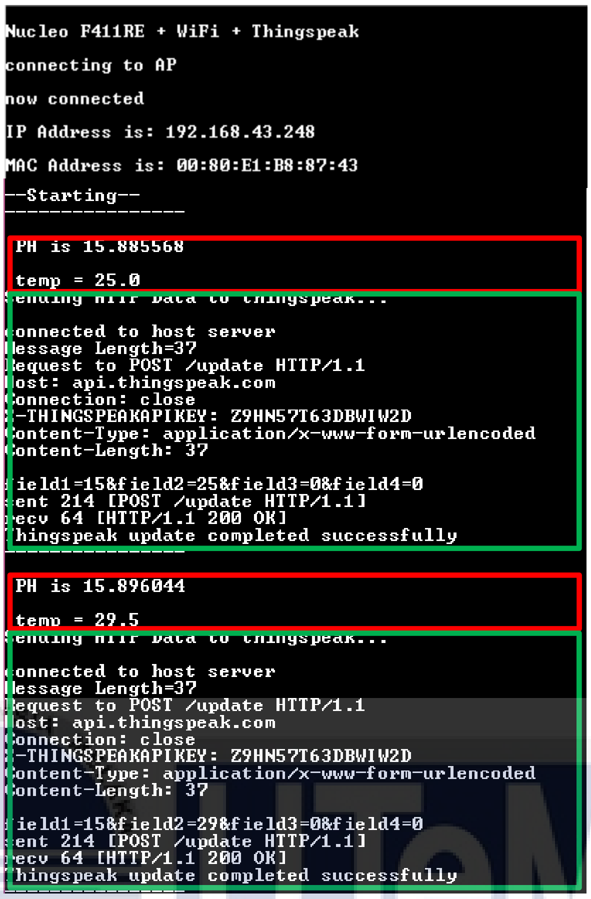
The setup of SPWF01SA Wi-Fi microchip, pH sensor, and temperature sensor data after both sensors' probe dipped into a solution. The region of red box shows in Figure 4.10 is the setup of SPWF01SA by connecting to a Wi-Fi access point and the configuration of the IP address as well as MAC address. These setup process will respond by a result by a result which is either connected or not connected, the IP address of the connection of network that took part in while the MAC address of the microcontroller.



```
Nucleo F411RE + WiFi + Thingspeak

connecting to AP

now connected

IP Address is: 192.168.43.248

MAC Address is: 00:80:E1:B8:87:43
--Starting--
---------------

 PH is 15.885568

 temp = 25.0
Sending HTTP Data to thingspeak...

connected to host server
Message Length=37
Request to POST /update HTTP/1.1
Host: api.thingspeak.com
Connection: close
X-THINGSPEAKAPIKEY: Z9HN57T63DBWIW2D
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

field1=15&field2=25&field3=0&field4=0
sent 214 [POST /update HTTP/1.1]
recv 64 [HTTP/1.1 200 OK]
Thingspeak update completed successfully
```

**Figure 4.15 The monitoring interface in Tera Term (setup).**

The monitoring interface of the Tera Term in Figure 4.10 shows two cycles of the main function loop. The primary function loop begins with the data receive and the data conversion process. The collected data is printed out and displayed on decimal values. In the data conversion process, the pH and temperature data in the data field further processed by a mathematical formula of each parameter. The actual reading of sensors data after conversion printed out. Then, it continues with the data request process. A non-blocking function used to send the parameter IDs to request the real-time sensor data from it. With ESP8266 Wi-Fi microchip, lastly, the updated actual sensors data submitted to the cloud server by using an HTTP request. In Figure 4.10 below, the results of data receive & data conversion and data request & data stored in the cloud server process divided into the section with red and green boxes, respectively. The sensor parameter that requested in this project includes the pH level and temperature of a solution.



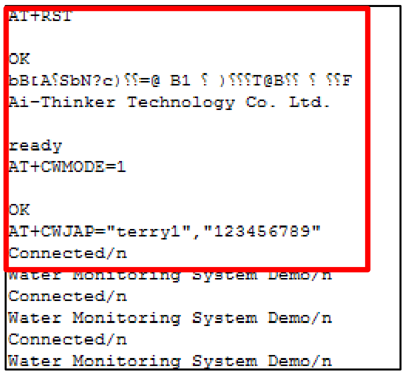
```
Nucleo F411RE + WiFi + Thingspeak
connecting to AP
now connected
IP Address is: 192.168.43.248
MAC Address is: 00:80:E1:B8:87:43
--Starting--
----------------
 PH is 15.885568

 temp = 25.0
Sending HTTP Data to thingspeak...

connected to host server
Message Length=37
Request to POST /update HTTP/1.1
Host: api.thingspeak.com
Connection: close
X-THINGSPEAKAPIKEY: Z9HN57T63DBWIW2D
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

field1=15&field2=25&field3=0&field4=0
Sent 214 [POST /update HTTP/1.1]
Recv 64 [HTTP/1.1 200 OK]
Thingspeak update completed successfully
----------------
 PH is 15.896044

 temp = 29.5
Sending HTTP Data to thingspeak...

connected to host server
Message Length=37
Request to POST /update HTTP/1.1
Host: api.thingspeak.com
Connection: close
X-THINGSPEAKAPIKEY: Z9HN57T63DBWIW2D
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

field1=15&field2=29&field3=0&field4=0
Sent 214 [POST /update HTTP/1.1]
Recv 64 [HTTP/1.1 200 OK]
Thingspeak update completed successfully
----------------
```

**Figure 4.16 The monitoring interface in Tera Term software along with receiving, converting, and storing data in the cloud.**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

#### 4.4.2.2 Arduino UNO R3

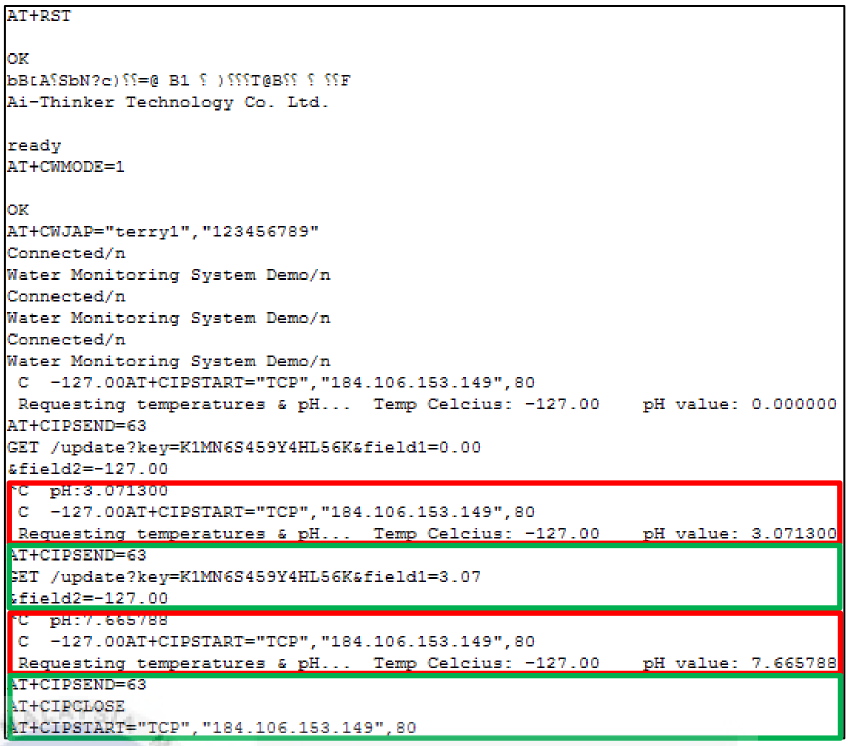
The setup of ESP8266 Wi-Fi microchip, pH sensor, and temperature sensor data after both sensors' probe dipped into a solution. The region of red box shows in Figure 4.11 is the setup of ESP8266 by connecting to a Wi-Fi access point by using the AT command. These setup process will respond by a result by a result which is either ok or failed to indicate the behavior of these devices when booting it up.



```
AT+RST

OK
bB[A⸮SbN?c)⸮⸮=@ B1 ⸮ )⸮⸮⸮T@B⸮⸮ ⸮ ⸮⸮F
Ai-Thinker Technology Co. Ltd.

ready
AT+CWMODE=1

OK
AT+CWJAP="terry1","123456789"
Connected/n
Water Monitoring System Demo/n
Connected/n
Water Monitoring System Demo/n
Connected/n
Water Monitoring System Demo/n
```

**Figure 4.17 The monitoring interface in Serial Monitor (setup).**

The monitoring interface of Serial Monitor in Figure 4.12 shows two cycles of the main function loop. The primary function loop begins with the data receive and the data conversion process. The received data printed out. In the data conversion process, the pH and temperature data in the data field further processed by a mathematical formula of each parameter. The actual reading of sensors data after conversion printed out. Then, it continues with the data request process. A non-blocking function to send the parameters to request the real-time sensor data from it. Lastly, the updated actual sensors data sent to the cloud server by using an HTTP request with ESP8266 Wi-Fi microchip. In Figure 4.12 below, red and green boxes represented the results of data receive & data conversion, data request & data storing in the cloud server process into the section with, respectively. The sensor parameter that requested in this project includes the pH level and temperature of a solution.



```
AT+RST

OK
bB(A⸮SbN?c)⸮⸮=@ B1 ⸮ )⸮⸮⸮T@B⸮⸮ ⸮ ⸮⸮F
Ai-Thinker Technology Co. Ltd.

ready
AT+CWMODE=1

OK
AT+CWJAP="terry1","123456789"
Connected/n
Water Monitoring System Demo/n
Connected/n
Water Monitoring System Demo/n
Connected/n
Water Monitoring System Demo/n
 C  -127.00AT+CIPSTART="TCP","184.106.153.149",80
 Requesting temperatures & pH...  Temp Celcius: -127.00    pH value: 0.000000
AT+CIPSEND=63
GET /update?key=K1MN6S459Y4HL56K&field1=0.00
&field2=-127.00
°C  pH:3.071300
 C  -127.00AT+CIPSTART="TCP","184.106.153.149",80
 Requesting temperatures & pH...  Temp Celcius: -127.00    pH value: 3.071300
AT+CIPSEND=63
GET /update?key=K1MN6S459Y4HL56K&field1=3.07
&field2=-127.00
°C  pH:7.665788
 C  -127.00AT+CIPSTART="TCP","184.106.153.149",80
 Requesting temperatures & pH...  Temp Celcius: -127.00    pH value: 7.665788
AT+CIPSEND=63
AT+CIPCLOSE
AT+CIPSTART="TCP","184.106.153.149",80
```

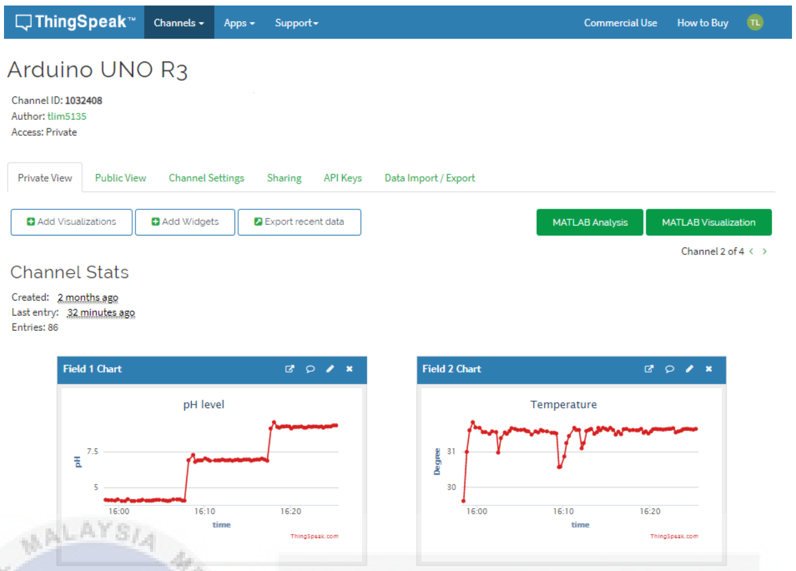
**Figure 4.18 The monitoring interface in Serial Monitor**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 4.5 Water Monitoring System in the ThingSpeak IoT platform.

To ensure this Water Monitoring System achieve the objective listed in this project, a series of data is taken by using this Water Monitoring System in a different water solution with different pH value. This Vehicle Monitoring System starts to collect and store the sensor data when once the microcontroller booted up. To compensate for the accuracy of the result, the first 30 samples as initializing of the water monitoring system. Then, 30 samples for acid, 30 samples for neutral, and 30 samples for alkaline. Hence, a total of 100 samples of data will be collected for the study and compare the system performance of microcontrollers. The actual sensor data are collected and sent to the ThingSpeak IoT platform along with this data collection with an average delay interval of about 15 seconds. Figure 4.6 and Figure 4.7 below shows the sensors data in three different water solution that presented in ThingSpeak

IoT platform user interface. These two sensors data are plotted vs. time in graphical form for better visibility.



Arduino UNO R3

Channel ID: 1032408
Author: tlim5135
Access: Private

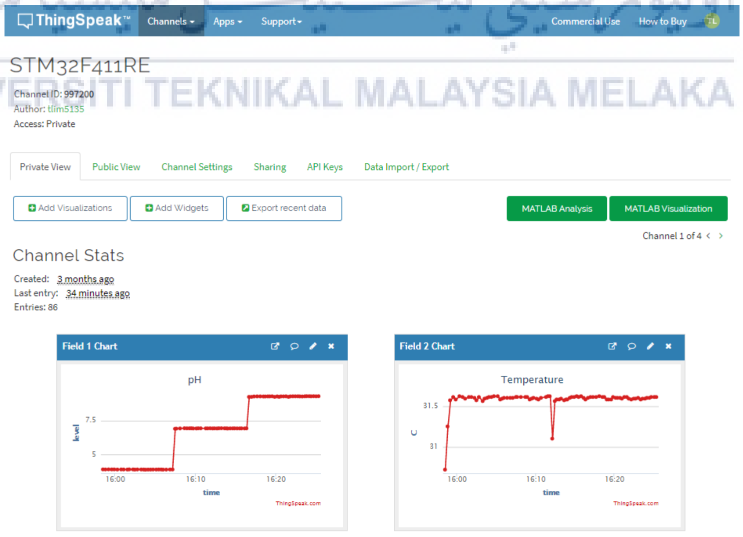
Channel 2 of 4

Channel Stats

Created: 2 months ago
Last entry: 32 minutes ago
Entries: 86

**Figure 4.19 Graphical User Interface in the ThingSpeak IoT platform (Arduino UNO R3).**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



STM32F411RE

Channel ID: 997200
Author: tlim5135
Access: Private

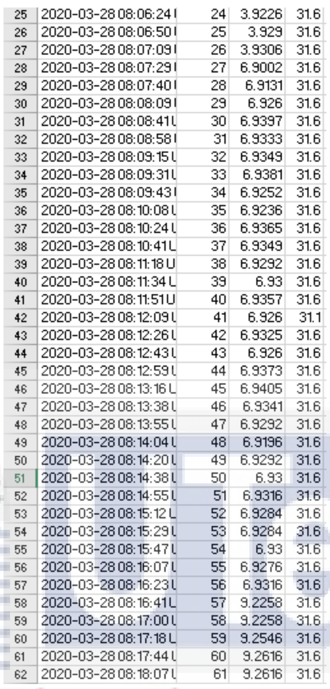
Channel 1 of 4

Channel Stats

Created: 3 months ago
Last entry: 34 minutes ago
Entries: 86

**Figure 4.20 Graphical User Interface in the ThingSpeak IoT platform (STM32F411RE).**

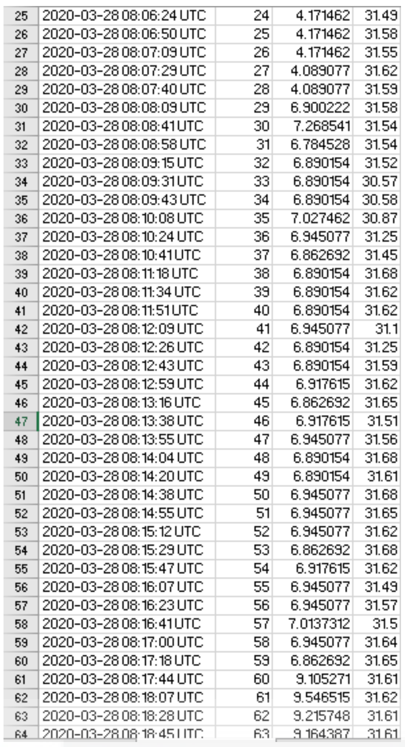
Other than observing the collected data in graphical form, users can also observe the collected data in tabular form. Users just need to simply download the raw data set in the data export function offered by ThingSpeak. Figure 4.8 below shows the tabulated raw data in Microsoft Excel.



| | | | | |
|---|---|---|---|---|
| 25 | 2020-03-28 08:06:24 U | 24 | 3.9226 | 31.6 |
| 26 | 2020-03-28 08:06:50 U | 25 | 3.929 | 31.6 |
| 27 | 2020-03-28 08:07:09 U | 26 | 3.9306 | 31.6 |
| 28 | 2020-03-28 08:07:29 U | 27 | 6.9002 | 31.6 |
| 29 | 2020-03-28 08:07:40 U | 28 | 6.9131 | 31.6 |
| 30 | 2020-03-28 08:08:09 U | 29 | 6.926 | 31.6 |
| 31 | 2020-03-28 08:08:41 U | 30 | 6.9397 | 31.6 |
| 32 | 2020-03-28 08:08:58 U | 31 | 6.9333 | 31.6 |
| 33 | 2020-03-28 08:09:15 U | 32 | 6.9349 | 31.6 |
| 34 | 2020-03-28 08:09:31 U | 33 | 6.9381 | 31.6 |
| 35 | 2020-03-28 08:09:43 U | 34 | 6.9252 | 31.6 |
| 36 | 2020-03-28 08:10:08 U | 35 | 6.9236 | 31.6 |
| 37 | 2020-03-28 08:10:24 U | 36 | 6.9365 | 31.6 |
| 38 | 2020-03-28 08:10:41 U | 37 | 6.9349 | 31.6 |
| 39 | 2020-03-28 08:11:18 U | 38 | 6.9292 | 31.6 |
| 40 | 2020-03-28 08:11:34 U | 39 | 6.93 | 31.6 |
| 41 | 2020-03-28 08:11:51 U | 40 | 6.9357 | 31.6 |
| 42 | 2020-03-28 08:12:09 U | 41 | 6.926 | 31.1 |
| 43 | 2020-03-28 08:12:26 U | 42 | 6.9325 | 31.6 |
| 44 | 2020-03-28 08:12:43 U | 43 | 6.926 | 31.6 |
| 45 | 2020-03-28 08:12:59 U | 44 | 6.9373 | 31.6 |
| 46 | 2020-03-28 08:13:16 U | 45 | 6.9405 | 31.6 |
| 47 | 2020-03-28 08:13:38 U | 46 | 6.9341 | 31.6 |
| 48 | 2020-03-28 08:13:55 U | 47 | 6.9292 | 31.6 |
| 49 | 2020-03-28 08:14:04 U | 48 | 6.9196 | 31.6 |
| 50 | 2020-03-28 08:14:20 U | 49 | 6.9292 | 31.6 |
| 51 | 2020-03-28 08:14:38 U | 50 | 6.93 | 31.6 |
| 52 | 2020-03-28 08:14:55 U | 51 | 6.9316 | 31.6 |
| 53 | 2020-03-28 08:15:12 U | 52 | 6.9284 | 31.6 |
| 54 | 2020-03-28 08:15:29 U | 53 | 6.9284 | 31.6 |
| 55 | 2020-03-28 08:15:47 U | 54 | 6.93 | 31.6 |
| 56 | 2020-03-28 08:16:07 U | 55 | 6.9276 | 31.6 |
| 57 | 2020-03-28 08:16:23 U | 56 | 6.9316 | 31.6 |
| 58 | 2020-03-28 08:16:41 U | 57 | 9.2258 | 31.6 |
| 59 | 2020-03-28 08:17:00 U | 58 | 9.2258 | 31.6 |
| 60 | 2020-03-28 08:17:18 U | 59 | 9.2546 | 31.6 |
| 61 | 2020-03-28 08:17:44 U | 60 | 9.2616 | 31.6 |
| 62 | 2020-03-28 08:18:07 U | 61 | 9.2616 | 31.6 |

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**Figure 4.21 Tabulated sensor raw data in Microsoft Excel (STM32F411RE).**



| | | | | |
|---|---|---|---|---|
| 25 | 2020-03-28 08:06:24 UTC | 24 | 4.171462 | 31.49 |
| 26 | 2020-03-28 08:06:50 UTC | 25 | 4.171462 | 31.58 |
| 27 | 2020-03-28 08:07:09 UTC | 26 | 4.171462 | 31.55 |
| 28 | 2020-03-28 08:07:29 UTC | 27 | 4.089077 | 31.62 |
| 29 | 2020-03-28 08:07:40 UTC | 28 | 4.089077 | 31.59 |
| 30 | 2020-03-28 08:08:09 UTC | 29 | 6.900222 | 31.58 |
| 31 | 2020-03-28 08:08:41 UTC | 30 | 7.268541 | 31.54 |
| 32 | 2020-03-28 08:08:58 UTC | 31 | 6.784528 | 31.54 |
| 33 | 2020-03-28 08:09:15 UTC | 32 | 6.890154 | 31.52 |
| 34 | 2020-03-28 08:09:31 UTC | 33 | 6.890154 | 30.57 |
| 35 | 2020-03-28 08:09:43 UTC | 34 | 6.890154 | 30.58 |
| 36 | 2020-03-28 08:10:08 UTC | 35 | 7.027462 | 30.87 |
| 37 | 2020-03-28 08:10:24 UTC | 36 | 6.945077 | 31.25 |
| 38 | 2020-03-28 08:10:41 UTC | 37 | 6.862692 | 31.45 |
| 39 | 2020-03-28 08:11:18 UTC | 38 | 6.890154 | 31.68 |
| 40 | 2020-03-28 08:11:34 UTC | 39 | 6.890154 | 31.62 |
| 41 | 2020-03-28 08:11:51 UTC | 40 | 6.890154 | 31.62 |
| 42 | 2020-03-28 08:12:09 UTC | 41 | 6.945077 | 31.1 |
| 43 | 2020-03-28 08:12:26 UTC | 42 | 6.890154 | 31.25 |
| 44 | 2020-03-28 08:12:43 UTC | 43 | 6.890154 | 31.59 |
| 45 | 2020-03-28 08:12:59 UTC | 44 | 6.917615 | 31.62 |
| 46 | 2020-03-28 08:13:16 UTC | 45 | 6.862692 | 31.65 |
| 47 | 2020-03-28 08:13:38 UTC | 46 | 6.917615 | 31.51 |
| 48 | 2020-03-28 08:13:55 UTC | 47 | 6.945077 | 31.56 |
| 49 | 2020-03-28 08:14:04 UTC | 48 | 6.890154 | 31.68 |
| 50 | 2020-03-28 08:14:20 UTC | 49 | 6.890154 | 31.61 |
| 51 | 2020-03-28 08:14:38 UTC | 50 | 6.945077 | 31.68 |
| 52 | 2020-03-28 08:14:55 UTC | 51 | 6.945077 | 31.65 |
| 53 | 2020-03-28 08:15:12 UTC | 52 | 6.945077 | 31.62 |
| 54 | 2020-03-28 08:15:29 UTC | 53 | 6.862692 | 31.68 |
| 55 | 2020-03-28 08:15:47 UTC | 54 | 6.917615 | 31.62 |
| 56 | 2020-03-28 08:16:07 UTC | 55 | 6.945077 | 31.49 |
| 57 | 2020-03-28 08:16:23 UTC | 56 | 6.945077 | 31.57 |
| 58 | 2020-03-28 08:16:41 UTC | 57 | 7.0137312 | 31.5 |
| 59 | 2020-03-28 08:17:00 UTC | 58 | 6.945077 | 31.64 |
| 60 | 2020-03-28 08:17:18 UTC | 59 | 6.862692 | 31.65 |
| 61 | 2020-03-28 08:17:44 UTC | 60 | 9.105271 | 31.61 |
| 62 | 2020-03-28 08:18:07 UTC | 61 | 9.546515 | 31.62 |
| 63 | 2020-03-28 08:18:28 UTC | 62 | 9.215748 | 31.61 |
| 64 | 2020-03-28 08:18:45 UTC | 63 | 9.164387 | 31.61 |

**Figure 4.22 Tabulated sensor raw data in Microsoft Excel (Arduino UNO R3).**

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

From the sensors data presented in graphical and tabular above, the pH values of acid are fallen within 3.917728 to 3.939487 and 4.0616 to 4.1891 for STM32F411RE and Arduino UNO R3, respectively. As for the benchmark for this acid pH value that should be obtained is 4.01 to 4.02 at the room temperature of 30℃ to 35℃. The pH value for neutral, the Arduino UNO R3 microcontrollers has measured the pH values which fall within 6.7845 to 7.2685. While for the STM32F411RE, it estimated the pH value falls that within the value of 6.900222 to 6.940512. Which the reference pH value should be within 6.85 to 6.84 at room temperature in between 30℃ to 35℃. Last but not least, for alkaline pH value, Arduino UNO has measured the pH value that falls within 9.1053 to 9.5465 while the STM32F411RE has measured the pH value that falls within 9.225815 to 9.275671.

For the benchmark pH value of alkaline, it should fall within the pH value of 9.14 to 9.10 at room temperature of 30℃ to 35℃.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# CHAPTER 5

# CONCLUSION AND FUTURE WORKS

This chapter will describe the conclusion and recommendation for the future development of this project.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 5.1 Conclusion

The first goal of this project is to compare and study Arduino UNO R3 and STM32F411RE microcontrollers' speed responsiveness with RISC and CISC instruction set. In this project, the comparative and study of Arduino UNO R3 and STM32F411RE speed responsiveness are using the techniques of latency and integer math comparison. During this study, hardware, as well as the software, were included. For the minimum latency and integer math comparison, the generated waveform of the operation, and then analyzed. From that particular waveform, the speed of each operation of integer math and minimum latency is being captured and observed. A reviewed to determine the speed responsiveness of each microcontroller. The analyzed data obtained is then tabulated to have an overview of each speed responsiveness. Form the result, it is evident that the STM32F411RE is faster than the Arduino UNO R3 compared to STM32F411RE. The proved result in sections 4.1 and 4.2.

The second goal of this project is to design and develop a cloud-based water monitoring system that can monitor data in real-time using the ThingSpeak IoT platform. Besides, with the real-time data uploaded to the IoT platform will be analyzed to determine the microcontroller wireless connection latency and stability. The technique or method to determine that characteristics are by using the time response analysis. In this project, the Water Monitoring System is implementing a real-time application that purposely to collect pH level and temperature reading of a solution with the help of pH sensors (SEN0161) and temperature sensor (DS18B20). The data of each parameter obtained is then sending to the cloud database in the ThingSpeak IoT platform. The user or authorities can inspect the data in real-time using the Graphical User Interface (GUI) implemented by ThingSpeak. The data is to download and analyze studies. From the survey, although both microcontrollers are stable. Still, the only difference is that from the perspective of upload speed, consistency, and efficient performance of a system, the STM32F411RE is once again faster than the Arduino UNO R3. The result of this function if proven in section 4.4.

In conclusion, it is a success, and all objectives mentioned are achieved in conducting this project of comparison and study of Arduino UNO R3 and STM32F411RE development board in a water monitoring system.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 5.2 Future Works

There are lots of ways to study and compare the microcontroller STM32F411RE. However, only minimum latency, integer math, and wireless performance comparisons were uses in this project. The wireless performance comparison brings into this project are purposely to apply or bring in the microcontroller into the real-life application to determine how well or how real the microcontrollers are potential to perform. The only parameters that tested in the project are pH level and temperature as to check how well the system performance of the microcontroller and perform with lesser parameters. The future development for this project can proceed by adding more parameters. With these, it can determine how well the system performance of microcontrollers was able to handle such substantial parameters. Besides, these additional parameters will significantly benefit the user, especially to the company that manages the water quality monitoring services and references for fishing industries. More types of parameters in a water monitoring system will give a more accurate result in indicating a vehicle or driver for the end-user. Besides, this project can be future enhance by modifying it to be a mobility device that able to move freely on the surface of the water to identify & tag by providing the parameters of the water so that it would ease the user for providing the neutralizer and water treatment services. Last but not least, compare more and different types of microcontrollers at the same time to give the real system performance of a microcontroller in the market to the consumer. Besides, it is purposely to have a clear and deep understanding to choose the optimum microcontroller for their projects, mainly students like us who took their Final Year Project.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

# REFERENCES

[1] S. Dhiman, “Performance Testing: A Comparative Study and Analysis of Web Service Testing Tools,” *Int. J. Recent Trends Eng. Res.*, vol. 4, no. 3, pp. 95–100, 2018, doi: 10.23883/ijrter.2018.4102.tbuwk.

[2] G. Gridling and B. Weiss, “Microcontrollers,” *Introd. to Microcontrollers*, pp. 379–414, 2007, doi: 10.1016/b978-012451838-4/50016-9.

[3] M. K. Parai, B. Das, and G. Das, “An Overview of Microcontroller Unit: From Proper Selection to Specific Application,” *Int. J. Soft Comput.*, no. 6, pp. 228–231, 2013, doi: 2231-2307.

[4] R. Khadse, N. Gawai, and B. M. Faruk, “Overview and Comparative Study of Different Microcontrollers,” [Online]. Available: www.ijraset.com.

[5] P. R.Dinkir, Patnaik, “A Comparative Study of Arduino, Raspberry Pi and ESP8266 as IoT Development Board,” *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 5, pp. 2350–2352, 2017.

[6] M. Fadzli Abdul Shaib, R. Abdul Rahim, S. Z. M. Muji, N. Shima, and M. Z. Zawahir, “Comparison between two different types of microcontroller in

developing optical tomography controller unit," *J. Teknol. (Sciences Eng.*, vol. 64, no. 5, pp. 13–17, 2013, doi: 10.11113/jt.v64.2126.

[7] Y. Güven, E. Coşgun, S. Kocaoğlu, H. Gezici, and E. Yilmazlar, "Understanding the Concept of Microcontroller Based Systems To Choose The Best Hardware For Applications," *Res. Inven. Int. J. Eng. Sci.*, vol. 7, no. December, p. 38, 2017.

[8] R. Khadse, N. Gawai, and B. M. Faruk, "Overview and Comparative Study of Different Microcontrollers," vol. 2, no. Xii, pp. 311–315, 2014.

[9] M. Kumar Jha, R. Kumari Sah, M. S. Rashmitha, R. Sinha, B. Sujatha, and K. V. Suma, "Smart Water Monitoring System for Real-Time Water Quality and Usage Monitoring," *Proc. Int. Conf. Inven. Res. Comput. Appl. ICIRCA 2018*, no. Icirca, pp. 617–621, 2018, doi: 10.1109/ICIRCA.2018.8597179.

[10] R. Gouws and A. S. Nieuwoudt, "Design and cost analysis of an automation system for swimming pools in South Africa," *Proc. 20th Conf. Domest. Use Energy, DUE 2012*, no. April 2012, pp. 9–15, 2012.

[11] D. E. Bolanakis, A. K. Rachioti, and E. Glavas, "Nowadays trends in microcontroller education: Do we educate engineers or electronic hobbyists? Recommendation on a multi-platform method and system for lab training activities," *IEEE Glob. Eng. Educ. Conf. EDUCON*, no. April, pp. 73–77, 2017, doi: 10.1109/EDUCON.2017.7942826.

[12] Farnell, "Arduino Uno Datasheet," *Datasheets*, pp. 1–4, 2013.

[13] STMicroelectronics, "Arm® Cortex®-M4 32b MCU+FPU, 125 DMIPS,

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

512KB Flash, 128KB RAM, USB OTG FS, 11 TIMs, 1 ADC, 13 comm. interfaces," no. December, 2017.

[14] MAKER.IO STAFF, "Powerful Low-Cost Arduino Alternatives: STM32 Nucleo," 2016. https://www.digikey.my/en/maker/blogs/st-nucleo-a-powerful-low-cost-alternative-to-the-arduino (accessed Nov. 29, 2019).

[15] L. Louis, "Working Principle of Arduino and Using It As a Tool for Study and W Orking P Rinciple of a Rduino and U Sing I T," no. July, 2018, doi: 10.5121/ijcacs.2016.1203.

[16] I. Tsekoura, G. Rebel, P. Glosekotter, and M. Berekovic, "An evaluation of energy efficient microcontrollers," *2014 9th Int. Symp. Reconfigurable Commun. Syst. ReCoSoC 2014*, no. May, 2014, doi: 10.1109/ReCoSoC.2014.6861368.

[17] A. Ng, J. Lepinski, D. Wigdor, S. Sanders, and P. Dietz, "Designing for Low-Latency Direct-Touch Input," pp. 453–464, 2012.

[18] J. Deber, R. Jota, C. Forlines, and D. Wigdor, "How much faster is fast enough? User perception of latency & latency improvements in direct and indirect touch," *Conf. Hum. Factors Comput. Syst. - Proc.*, vol. 2015-April, no. 1, pp. 1827–1836, 2015, doi: 10.1145/2702123.2702300.

[19] "Performance comparison: Arduino Uno vs. ST Nucleo L152RE | Mbed." https://os.mbed.com/users/iliketux/notebook/performance-comparison-arduino-uno-vs-st-nucleo-l1/ (accessed Nov. 29, 2019).

[20] G. Simoes, C. Dionisio, A. Gloria, P. Sebastiao, and N. Souto, "Smart System

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

for Monitoring and Control of Swimming Pools," *IEEE 5th World Forum Internet Things, WF-IoT 2019 - Conf. Proc.*, no. 1, pp. 829–832, 2019, doi: 10.1109/WF-IoT.2019.8767240.

[21] M. Kit and S. K. U. Sen, "PH meter(SKU: SEN0161)."

[22] Maximintegrated, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer DS18B20 Programmable Resolution 1-Wire Digital Thermometer Absolute Maximum Ratings," vol. 92, pp. 1–20, 2019.

[23] S. Mccaffrey, "Water Quality Parameters & indicator," pp. 0–3, 1997.

[24] C. R. Shah, "Which Physical , Chemical and Biological Parameters of water determine its quality?," *ResearchGate*, no. June, pp. 1–74, 2017, doi: 10.13140/RG.2.2.29178.90569.

[25] "bildr » One Wire Digital Temperature. DS18B20 + Arduino." https://bildr.org/2011/07/ds18b20-arduino/ (accessed May 14, 2020).

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

[26] B. Sigdel, "Water Quality Measuring Station," p. pp 1-37, 2017.

[27] J. J.Barron, C. Ashton, and L. Geary, "The Effects of Temperature on pH Measurement," *Ecol. Reprod. Wild Domest. Mamm.*, pp. 1–7, doi: 10.1007/978-94-011-6527-3_13.

[28] "pH value, index." http://www.soz-etc.com/med/pH/pH-index-ENGL.html (accessed May 14, 2020).

[29] J. Gowthamy, C. R. Reddy, P. Meher, S. Shrivastava, and G. Kumar, "Smart Water Monitoring System using IoT," *Int. Res. J. Eng. Technol.*, p. 1170, 2008,

[Online]. Available: www.irjet.net.

[30] K. Karimi and K. Salah-ddine, "a Comparative Study of the Implementations Design for Smart Homes / Smart Phones Systems," vol. 7, no. January, pp. 141–149, 2018.

[31] N. Thirupathi Rao, D. Bhattacharyya, V. Madhusudhan Rao, and T. H. Kim, "Water quality testing and monitoring system," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 5, pp. 162–166, 2019.

[32] A. N. Prasad, K. A. Mamun, F. R. Islam, and H. Haqva, "Smart water quality monitoring system," *2015 2nd Asia-Pacific World Congr. Comput. Sci. Eng. APWC CSE 2015*, no. December 2016, 2016, doi: 10.1109/APWCCSE.2015.7476234.

[33] A. N. Prasad, K. A. Mamun, F. R. Islam, and H. Haqva, "Smart water quality monitoring system," in *2015 2nd Asia-Pacific World Congress on Computer Science and Engineering, APWC on CSE 2015*, May 2016, doi: 10.1109/APWCCSE.2015.7476234.

[34] S. Geetha and S. Gouthami, "Internet of things enabled real time water quality monitoring system," *Smart Water*, vol. 2, no. 1, pp. 1–19, 2016, doi: 10.1186/s40713-017-0005-y.

[35] E. Systems, "ESP8266EX," 2019.

[36] A. Uno *et al.*, "X-NUCLEO-IDW01M1 Wi-Fi expansion board based on SPWF01SA module for STM32," pp. 1–8, 2016.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

[37] K. K. Patel, S. M. Patel, and P. G. Scholar, “Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application &amp; Future Challenges,” *Int. J. Eng. Sci. Comput.*, vol. 6, no. 5, pp. 1–10, 2016, doi: 10.4010/2016.1482.

[38] P. Ganguly, “Selecting the right IoT cloud platform,” in *2016 International Conference on Internet of Things and Applications, IOTA 2016*, 2016, pp. 316–320, doi: 10.1109/IOTA.2016.7562744.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA