Final Year Project Report

Bachelor of Engineering

Micro-weather Station

Student: Danli Yan

GUID: 2109716Y

1st Supervisor: Duncan Bremner

2nd Supervisor: Guolong Cui

2016-2017

Student Number: 2109716Y	Student Name: Danli Yan
Course Code: UESTC4006P	Course Name: INDIVIDUAL PROJECT 4
Name of Lecturer:	Name of Demonstrator:
Title of Assignment: BEng Individual Project Report	

Declaration of Originality and Submission Information

I affirm that this submission is all my own work in accordance with the University of Glasgow Regulations and the School of Engineering requirements Signed (Student) :	UESTC4006P
Date of Submission: 28/4/2017	
Feedback from Lecturer to Student – to be completed by Lecturer or Demonstrator	
Grade Awarded: Feedback (as appropriate to the coursework which was assessed):	
Lecturer/Demonstrator:	Date returned to the Teaching Office:

Abstract

This project is the development of a self-contained weather station capable of monitoring several environmental conditions including temperature, relative humidity, light intensity and air quality. The device pre-processes this information and uploads the data to a cloud based database as well as a smart phone application. The data is also displayed on a local screen and stored in local memory for at least one month. Also, the working device is able to be recharged by solar power energy. The final prototype shows a great performance in monitoring surrounding environment with acceptable measurement error.

Index Terms— environmental monitoring, wireless communication, data analysis, weather station, cloud server

Acknowledgement

I would like to take this opportunity to thank professor Duncan for all his help and advice and the staff of Qinglian cloud for guiding me through the process of connecting to cloud server.

Table of Contents

List of Figures

List of Tables

1 Introduction

From ancient times, weather monitoring brings great convenience to people's life and production. For example, during the Qin and Han Dynasties, Chinese people created the 24 Solar Terms according to the law of the weather changing to guide farmers make time for cultivation [1]. In modern society, people depend on meteorological data and weather forecasts to determine activities such as travel, clothing and so on. Additionally, meteorological data detection is increasingly demanded in recent years with the rapid evolution of renewable energy sources as the performance of the system mostly relies on the weather conditions of where it is installed [2]. Weather monitoring generally includes common weather data such as temperature, relative humidity, wind speed, wind direction and atmospheric pressure. With the acceleration of urbanisation, the air pollution index has also gradually become one of the most important meteorological data for citizens [3]. Weather station is the equipment used to measure climate information which always composed by a collector, a processor and a transmitter. The collector collects weather-related measurements, the processor contributes to processing or computing and the transmitter finally sends these data to the receiving terminal's database for statistics and analysis [4]. Furthermore, considering the power supply for some remote areas is unavailable, weather stations usually have batteries rechargeable by solar power to ensure longer autonomous operation [3]. However, due to the high cost of installation and maintenance of such a system, weather stations are installed apart by a long distance. [5] Thus, weather forecast usually only tells people about a city’s weather condition of a certain period, not accurate for a specific house or company [6]. In addition, the traditional weather station is too large and expensive to be employed by home users or small factories for environmental monitoring.

Several attempts for constructing a cost-effective and small-scale environmental monitoring system have been performed by previous researchers. Adnan Shaout et al [6] presented an embedded design of low cost weather station with an ability to measure wind chill temperature and dressing index. The data is local displayed and transmitted to computer via serial port communication with wires, which means that the information is only accessible when the device connected with computer. A wireless transmission of climate data is employed by M. Benghanem [4], he used a microcontroller-based unit to record interested signals, while the data were transmitted to a base station, which was connected to computer for storage and further processing. Such systems are inconvenient when the required data must be distributed to several remote users [2]. It is possible that remote users access all the real-time data through the Internet. Therefore, S. Rosiek and F.J. Batlles [5] developed a data-acquisition system by using GSM (Global System of Mobile communication) technology as the mean of wireless transmission to allow the connection between the weather station and internet, an external memory was also equipped to keep local data up to 12 days. However, this research did not extend the study about the online storage and observation. A better experiment is performed by Abdullah Kadri et al [3], with the improvement of IOT (Internet of Things) techniques, they devised a cloud-based weather station which transmits data to a cloud server via an internet connected on-board GSM module, but the construction of cloud server is too complicated to realize for developers. Plus, because these researchers paid a lot of attention on the build of cloud, this prototype did not support some other basic functions like local display and external storage.

In fact, except for GSM, Wi-Fi offers an alternative approach to connect smart objects into the Internet. As the member of wireless communication, Wi-Fi is not feasible enough and has restricted coverage rank of the order 200 meters compared with other technologies [5]. However, it provides high speed of transmission and simple implementation which can be easily accessed by most ordinary people. Additionally, there are many provided cloud server based on a Wi-Fi module, making it more convenient and reliable to store and manage data online.

This report presents a low cost, small volume and solar powered weather station being capable of sensing some basic meteorological data and uploading it to a databased cloud so that the users could access real-time data whenever they log in the cloud through phones or computers. This micro-weather station also provides a user friendly local display and a large local storage room. In this report, I first describe the details about the method of design, then a discussion is followed behind to talk about the problem aroused during the project. After that, the results of testing are presented to evaluate the performance of the device. Finally, it draws a conclusion and provides suggestions for future work.

2 Project Implementation

The complete unit is supposed to have following features:

- collect meteorological data and store in local memory.
- upload data to cloud database.
- powered by solar energy.

Once powered by solar battery, the device begins to sense climate information and record the present time. These measurements are processed and then displayed on a LCD with a period of 10 seconds and transmitted to a cloud database one by one with the same frequency. Also, there is an external memory with 2GB storage space added to the controller to store processed local data for at least a month. The whole process can be divided into four main parts: local data acquisition, local data processing, upload to cloud and power supply. Figure 1 and Figure 2 show hardware design and software flow chart of the complete system. And the main code can be viewed in **[Appendix 1]**.

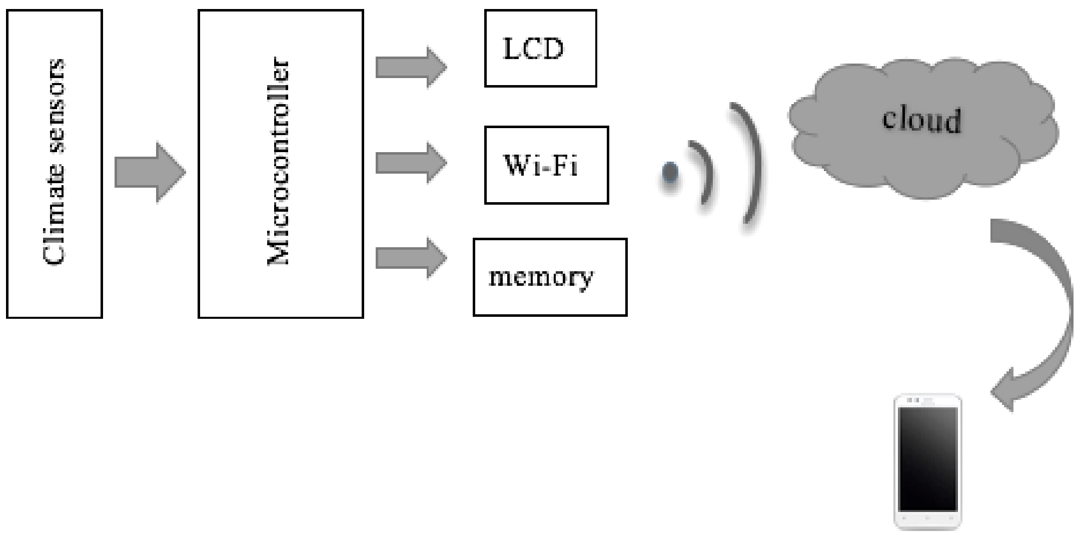


Figure 1. The block diagram of hardware design

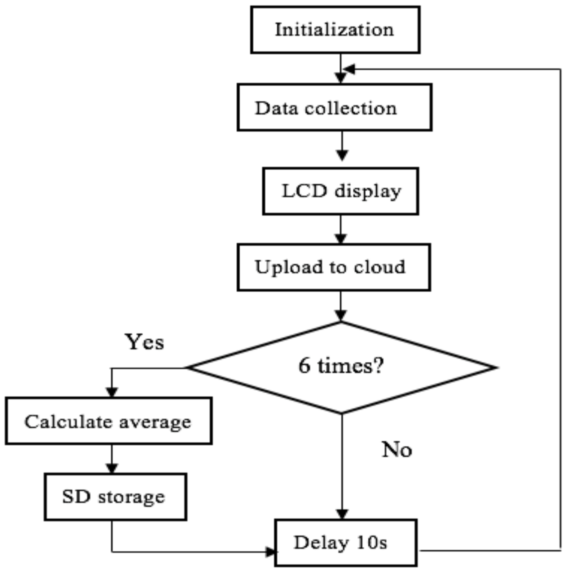


Figure 2. Flowchart illustrating the complete process of collecting, processing, uploading and storing climate data

2.1 Local data acquisition

The data acquisition system of the station consists of several climate sensors and a microcontroller. The microcontroller collects the output value of each sensor one by one and stores them in an array to facilitate further processing. Figure 3 shows the flow chart of the data collection program. Following context describes the hardware implementation and software design of each part of the system.

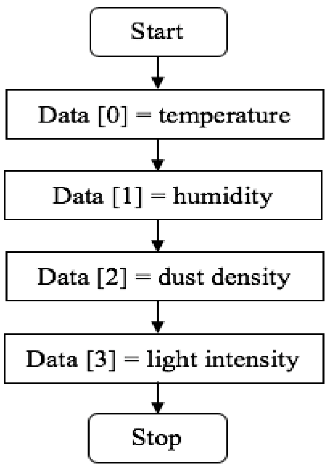


Figure 3. Flowchart of climate data collection, where variable “data” is an array to store four parameters.

2.1.1 Microcontroller

As a board widely used by engineers, Arduino Mega 2560 is chosen to be the MCU. One reason is that there is rich online resource as well as existed cloud servers based on it. With 256KB internal flash memory and large SRAM space of 8KB, it could store more instructions. Since most of the elements operating at 5V or 3.3V power supply, Arduino mega 2560 also provides these two types of output voltage to avoid voltage conversion. In addition, it has a full set of I/O interfaces including I2C, UART, SPI etc. to support different types of communication with sensors and modules [7].

2.1.2 Temperature and relative humidity sensing

The detection of temperature and relative humidity is dedicated to DHT11 sensor. It is designed to measure temperature from 0℃° to 50℃° degree and relative humidity from 20% to 90 %. The limitation lies in the measurable range, it cannot detect the temperature below 0℃°, but it satisfies the requirements of most situations in our daily life. Also, its advantage of sensing both temperature and humidity simultaneously overweighs this drawback as it saves pins of connection and simplifies the code. This sensor collects the information of temperature and humidity by outputting a series of digit numbers. A complete data transmission is 40 bits formed by 16 bits' humidity data, 16 bits' temperature data and 8 bits' checksum, so the user can extract corresponding digits to obtain humidity or temperature value.

2.1.3 Dust density sensing

A. Introduction of AQI

Air quality is a new environment condition that people are very concern with in recent years, especially for those live in urban areas. Poor air quality is possible to endanger respiratory system and even cause fatal decease such as cancers. AQI (Air Quality Index) is a standard used to indicate the air quality status and plays a significant role in analysing and evaluating a short-term variation of air pollution of a city [8]. According to the degree of contamination, it is classified into six stages: excellent, good, slight pollution, moderate pollution, severe pollution and serious pollution [9]. In this project, I roughly evaluated the value of AQI by measuring the dust density. Table 1 shows the different levels of AQI and corresponding dust density, the higher AQI index is, the terrible the air quality would be. As long as we get the value of dust density, we can also predict the level of AQI as well as the pollution degree. A table contains more information can be seen in **[Appendix 2]**.

Table 1. Classification of dust density and AQI where different colour present different level (Source: [10]. *Technical Regulation on Ambient Air Quality Index, 2012*, Ministry of Environmental Protection of the People's Republic of China, pp:2-4.)

Dust density (ug/m^3)	**AQI index range**	**Air quality level**	**Air quality status**
0~35	0~50	1	excellent
35~75	51~100	2	good
75~115	101~150	3	slight pollution
115~150	151~200	4	moderate pollution
150~250	201~300	5	severe pollution
250~500	>300	6	serious pollution

B. Dust sensor

Sharp GP2Y1010AU0F is a dust sensor which is composed by an infrared emitting diode (IRED) and a phototransistor [11]. This dust sensor is much cheaper and can detect the reflected light of dust in air. Especially, it is effective to detect very fine particle like the cigarette smoke. The drawback is that it could not detect PM2.5(fine particles with a diameter of 2.5μm or less) accurately, but just calculate an approximate value of PM2.5 according to a given equation. As Figure 4 (left) indicated, the IRED is driven in a period of 10ms, 0.32ms for on state and the rest time for off state. During the on state, the IRED detects the reflection of lights resulted by the dust pass through the hole and then outputs changeable pulse. From Figure 4 (right), we can see the value of output increases until reaches a peak at 0.28ms after the IRED is turned on. Thus, attention should be paid to sampling the detected signal at 0.28ms after the IRED emission to get a reliable measurement. Figure 5 shows the relationship between output voltage of the sensor and dust density, it is linear over the range of 0 to 0.5mg/m3. According to the graph, we can get an equation as

$$dust\ density(ug/m^3) = (0.17 \times output\ voltage(V) - 0.1) \times 1000 \qquad (2\text{-}1)$$

It should be noted that this equation is only established when the output voltage is between 0.9 V and 3.4 V, so we need regulate those outrange measurements first: if the output voltage is larger than 3.4 V, set it to be 3.4 V and if the output voltage is smaller than 0.9V, set it to be 0.9V. Afterwards, the corresponding level of air quality is possible to be obtained by using equation (2-1) .

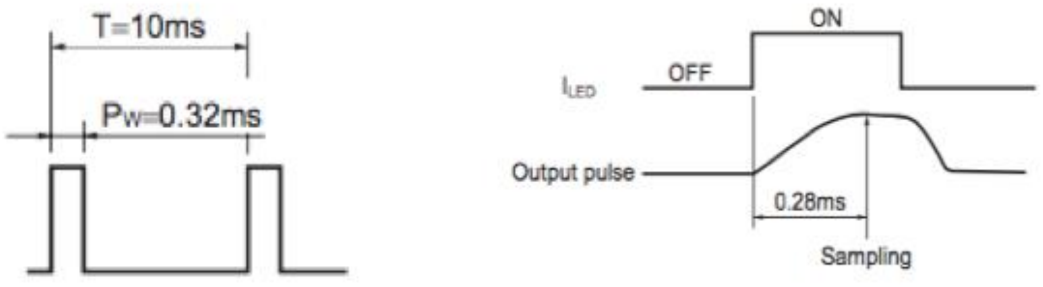


Figure 4. Pulse driven wave form (left) Sampling Timing of Output Pulse (right) (Source:[11]. *Application note of Sharp dust sensor GP2Y1010AU0F,* SHARP,2006, pp.1-9.)

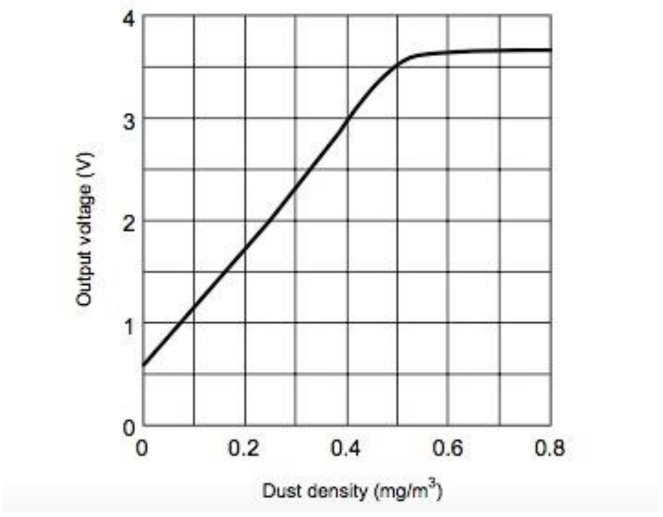
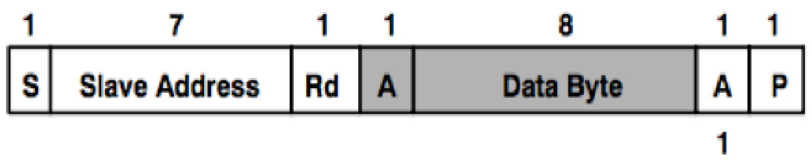


Figure 5. Output Voltage vs. Dust Density for Sharp GP2Y1010AU0F(source:[11]. Application note of Sharp dust sensor GP2Y1010AU0F, SHARP,2006, pp.1-9.)

2.1.4 Light intensity sensing

The measurement of luminance is usually used to maximize crop yields in agricultural production by determining the optimum light intensity for crop growth. TSL2561 is an optical digitizer that converts light intensity to 16 bits' digital signal output capable of communicating with microcontroller with I2C interface [12]. The transmission of light intensity information from slave (TSL2561) to master (Arduino mega 2560) is shown in Figure 6, the master dresses the specific slave and sets master-to-slave read mode, the slave then responds with acknowledge followed by the data byte which contains the key information, the master finally reads the data and transforms it into light intensity.



A	Acknowledge (this bit position may be 0 for an ACK or 1 for a NACK)
P	Stop Condition
Rd	Read (bit value of 1)
S	Start Condition
Sr	Repeated Start Condition
Wr	Write (bit value of 0)
X	Shown under a field indicates that that field is required to have a value of X
...	Continuation of protocol
□	Master-to-Slave
▨	Slave-to-Master

Figure 6. I2C Receive Byte Protocol (Source: [12]. TSL2560, TSL2561 LIGHT-TO-DIGITAL CONVERTER, Texas Advanced Optoelectronic Solutions Inc,2007, pp.1-38.)

2.1.5 Real-time accessing

RTC (Real Time Clock) is an essential section of weather station to generate accurate time information. the employment of a real-time clock module PCF8563 enables the acquisition of the information including year, month, day, weekday, hours, minutes and seconds based on 32.768 kHz quartz crystal with low power consumption [13]. Like the light intensity sensor, this module also talks to microcontroller with I2C interface. Fortunately, there are two on-board I2C ports allowing their connection at the same time. Once the time of beginning is set, this module will continue to record the time second by second based on the initial setting all the time, because there is an on-chip battery allowing the clock to keep working even though it is unconnected to the controller.

2.2 Local data processing

Data processing system includes local display and local storage. Although the micro-weather station senses the weather condition in a very small time interval, it is unnecessary to store the sensed values in such a high frequency, which may lead to the waste of memory space. To increase the reliability and accuracy of transferred data, initial calculation is required to compute an average value of several measurements. In this project, the period of measuring process is in the order of 10s, so I designed an algorithm aiming to derive the average value of 6 samples, which means the SD card could catch the information with a period of one minute. Figure 7 shows the calculation function of average value. The variable "i" is used to decide which parameter is calculated. For example, if i=0, this function outputs the average value of temperature. In this way, the average value of each parameter can be obtained by setting the value of "i" from 0 to 3.

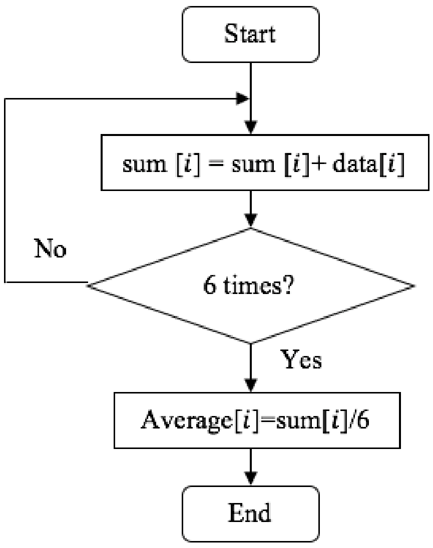


Figure 7. Function calculating the average value of every 6 measurements, where "sum", "data" and "average" are array to store the information of 4 parameters

2.2.1 LCD local display

After the controller accessing all the weather-related information, these dada is supposed to displayed for easier observation. one approach of monitoring the real-time outputs is to apply a serial port tool. However, this method does not work when the device has been completed and installed in a fixed place because the tool is a computer software which is impossible to monitor the device all the time. LCD local display has solved this problem; it provides developers a direction for correction if there is any problem with the sensor. For example, if the temperature value shown on the screen is unusual, I would firstly check the temperature sensor instead of wasting time on other sensors. Also, an on-device monitor helps some users grasp the operation of device and weather condition conveniently, especially for old people who dislike using phones and computers see the online data.

The implementation of LCD 12864 realized the function of local display. With 70.7×38.8 mm screen, it has a large field of vision which can show more climate data. In addition to that, this LCD offers two types of connection mode：8-bit parallel connection and SPI connection. In SPI communication, the LCD acts as a slave and receives instructions from master, which is Arduino mega 2560 in this project. It occupies only 3 digital pins on the MCU, much less than 8 pins required by parallel connection.

Another advantage of LCD 12860 is that the backlight brightness can be adjusted according to user intention. For this device, I added a switch to turn on and off the backlight, avoiding wasting of power when there is no need to display. The hardware connection is shown in Figure 8, Vcc is the power supply and PSB the pin controls the backlight. The voltage applied to PSB affects the brightness of screen and the higher the voltage is, the brighter the screen would be. Thus, when the switch is off, the LCD backlight on and vice visa. The backlight off does not have an influence on the operation of LCD, the data still shown on the screen but can hardly be seen.

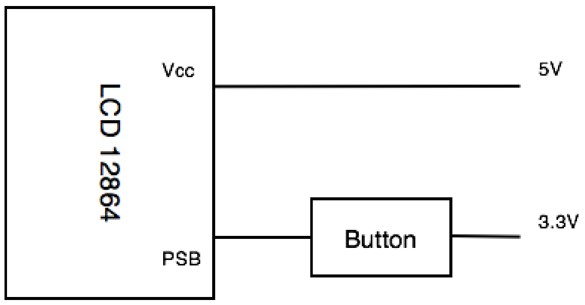
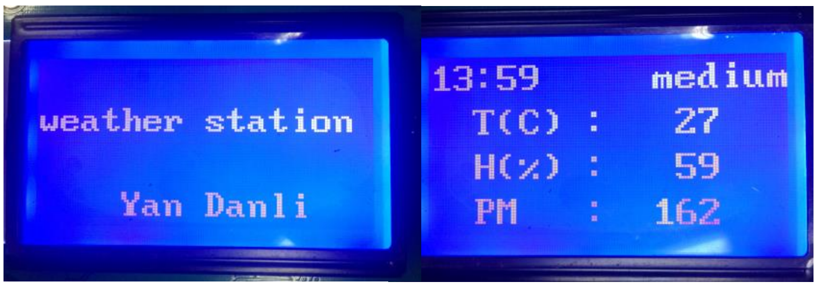


Figure 8. Hardware connection for backlight control of LCD

The frequency of refreshing the screen is determined to be 0.1HZ. If the frequency is too fast, it may cause clock disorder and lead to messy code shown on the screen. If the frequency is too slow, some transient change could not be observed in time and the time displayed on the screen may not right. For example, assuming that the screen refreshes every 2 minutes, the time shown on the screen would remain constant in next two minutes. That is also the reason why the second is not shown on the screen, showing hour and minute lets the real-time clock seems to be continuous. As for page display, two kinds of models are designed as shown in Figure 9 The left one is appeared when the LCD is initialized and lasts for about 5 seconds, it shows the device name and username. The other one displays the values of climate data, present time and pollution level, it works by following the program flowchart of Figure 10.



weather station

Yan Danli

```
13:59      medium
  T(C) :     27
  H(%) :     59
  PM   :    162
```

Figure 9. start-up screen (left) working screen (right).

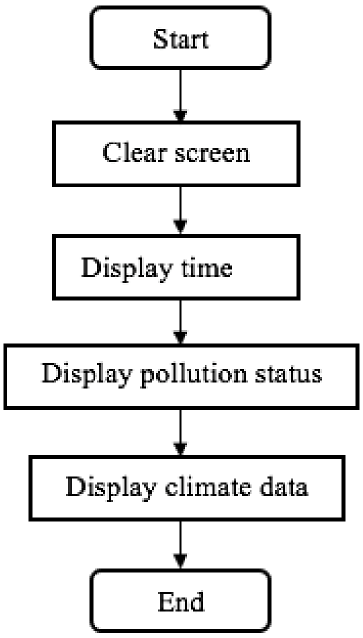


Figure 10. The flow chart of LCD display function

2.2.2 Local storage

Additional process of allocated data is to store them in the local memory for at least one month. This is necessary for some remote areas with fast changes in weather condition because variation is supposed to be recorded periodically. Unlike real-time information valued by ordinary users, scientists and meteorologists attach more importance on historical records for further analyses [14]. For instance, modern weather forecast system is based on the climate data over last decade to give predictions. The deployment of local storage is also an advanced action in response to a possible failure of wireless transmission. Furthermore, by using local storage, it would be possible to show averages on the local display for stand-alone application.

Due to the insufficient memory space of micro-controller, I have used an external SD card with 2GB storage room to save monthly weather measurements._The key point of choosing a suitable SD card is the volume of data, since the data is stored in local memory every minute, we can roughly estimate the data volume of a month based on a mall range of recordings. Through implementation **[Appendix 3]**, the hourly data volume is 2564 bytes, occupying around 3 KB storage space. Thus, the memory of micro SD card should be no less than $3KB\times24\times30 = 2.1MB$, which indicates that a 2GB card is capable of a Long-term data storage.

Concerning the storage format, each parameter value is separated by a comma so that it can be easily transferred to Excel for line graph plotting. A single sequence containing one-minute climate information looks like:

20.67,68.33,53.00,703.00,04/15/2017,00:01

Table 1 shows the explanation of sequence above.

Table 2. The meaning of the stored message

Code	Meaning
20.67	The temperature is 20.67 °C
68.33	The humidity is 68.33%
53.00	The dust density is $53.00ug/m^3$
703.00	The light intensity is $703.00lux$
04/15/2017	The date of the transmission is 2017-04-15
00:01	The hour of the average 00:01AM

2.3 Upload data to cloud

As a result of limited local memory space, transmitting data wirelessly to exterior world is a demanding task. However, most of previous methodologies only allow the processed information goes to a main station via antennas or has no ability to store the received data. In recent years, with the rapid development of the Internet of Things, the main method is to access embedded devices to the Internet for real-time monitoring data even remote control of equipment. There are two ways to enter the network, one through a specific website display, the second is based on the existing Internet of Things cloud platform to observe and store device data [15]. Here, I chose to upload weather data to cloud based database called Qinglian cloud as well as a phone application via Wi-Fi module every 10 seconds, providing users with real-time observation and secure storage services. The period of uploading determined by an internal timer packer of MCU, it is chosen to be 10 seconds because I want to compare the real-time data shown on the cloud with the average data stored in SD card. It helps me to see whether the average data variation could present the real change of information.

2.3.1 Wireless transmission

As a bridge between device and cloud server, Wi-Fi module is responsible to receive local data through a serial port and upload it into cloud server with the help of constructed SDK (Software Development Kit) [16]. Esp8266 developed by ATlink company is one of the partners of Qinglian cloud, so it provides developers with the relevant SDK of this module, which includes the library and specific protocol to connect the microcontroller to the Wi-Fi module. therefore, developers are able to quickly construct IOT concept of the device and establish real-time long connection with cloud by downloading the SDK into Esp8266.

2.3.2 Cloud platform

Qinglian cloud is an organized and secure IOT cloud server aiming at providing developers a safe, simple method to connect their device to cloud without complex programming. it has following basic features:

- Strong safety certification

  "Safety first" is the basic research and development concept of Qinglian cloud. This feature is significant in this age of information, people begin to worry about information safety as it always brings them trouble if their private information is released. To ensure cloud information security, Qinglian cloud uses the industry-standard SSL (Secure Sockets Layer)/ TLS（Transport Layer Security） protocol, which is a security protocol that provides security and data integrity for network communications [17].

- Cloud Unified Device Management

  After the device is connected to the cloud, the last on-line time, WIFI module version, MCU version, running status, user behavior and so on can be viewed in real time in the console. Moreover, the received data from device will be stored in the cloud database and displayed in a clear plot with time as the x-axis and parameter as the y-axis as shown in Figure 11.

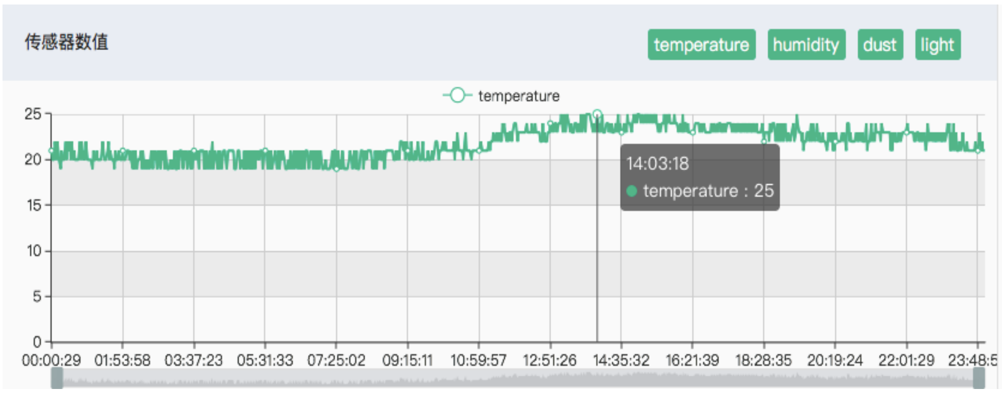


Figure 11. Cloud interface showing the temperature data obtained from device, where X- axis presents time and Y-axis presents the temperature

- Mobile phone application

  In addition to website display, the observation of real-time data is also achievable by a mobile application, you can even set an alarm using the application so that the cloud would send a message to remind users. Figure 12 shows the real-time display on LCD and screen and cellphone application.

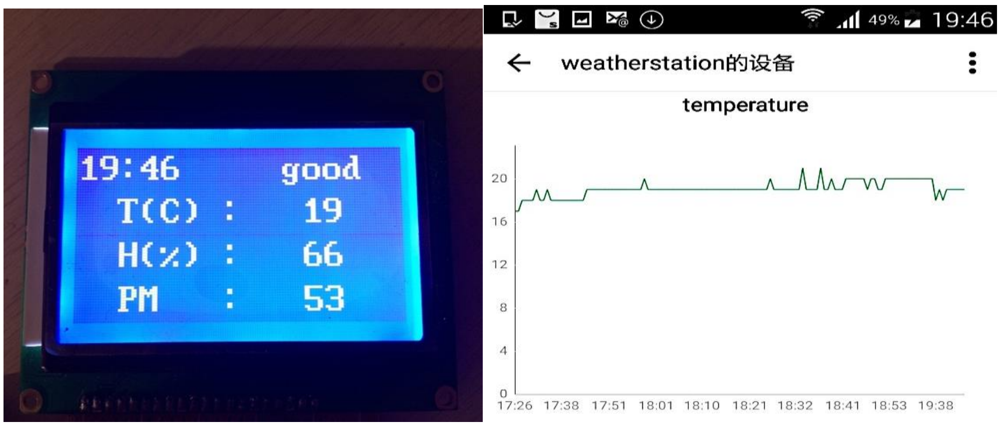
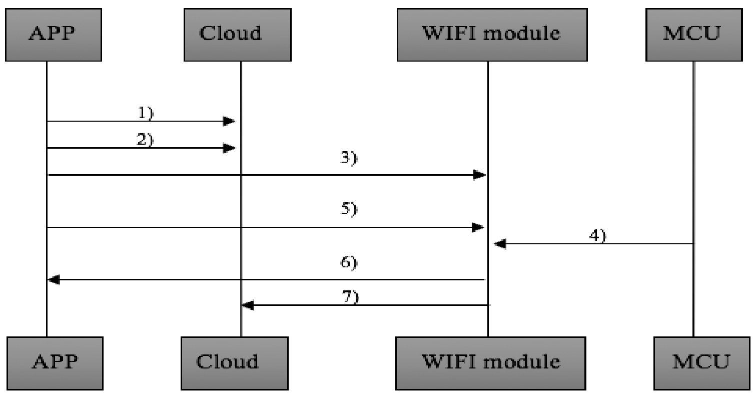


Figure 12. The real-time display of LCD (left) and a cropped screenshot of phone APP (right) indicating the APP can show the real-time data

The whole process of uploading data to Qinglian Cloud is shown in
Figure 13 Compilers build a private visual device in the cloud and define variables that they want to access from real device like temperature, humidity et. by observing specific rules. Then they will get a unique product ID and key to ensure connection with the corresponding real device. The match between the cloud platform and the real device is assisted by the phone application, eliminating the trouble to modify the program. For instance, if the Wi-Fi environment is changed, the users need only rematch the device by inputting new information of Wi-Fi in application.



1) register cloud
2) cloud device information
3) Wi-Fi information (name and passport)
4) register device
5) upload data through serial port
6) upload data
7) upload data

Figure 13. The process of publishing data on website and application

2.4 Power supply system

Considering the difficulty of accessing the main grid in remote areas, the system uses solar power, so that equipment becomes a fully self-contained, self-powered unit [2]. At the same time, the use of renewable energy is conducive to energy efficiency. Solar power supply system consists of three main components: solar panel, solar charge controller and storage battery. Power generation process is shown in Figure 9, the solar panel is capable of converting light energy into power energy, with the help of the controller, the generated power is then used for battery charge. In order to avoid the waste of light, the controller will monitor the battery power at any time. In the case of sufficient battery power, the solar panels charge the battery while charging the load. In the cloudy day or night, the battery discharges to ensure the normal operation of the equipment.

2.4.1 Storage battery

Storage battery is a type of rechargeable battery, it charges when the solar panel is working, and discharges when the output of panel is too small to power the load. Some characteristics of deployed battery is listed in Table 3.

Table 3. characteristics of storage battery taken from its user manual

Max charging voltage(V)	15
Max charging current(A)	2.1
Max discharging voltage (V)	12
Rated capacitance(AH)	7

In the choice of battery, voltage and battery capacity are the dominant considerations. Arduino mega 2560 can operate with an external supply of 7 to 20V in terms of its datasheet Therefore, the voltage of battery should be located at that range, which is chosen to be 12V here because of the limited types of battery. Regarding the capacity, it is determined by several parameters. Supposed the station is supplied by battery at least 20 hours and the working current of device is detected to be around 0.25A, then the least battery capacitance demanded is:

$$0.25A\times 20h = 5(A\cdot h) \tag{2-2}$$

2.4.2 Solar panel

A solar panel is a device that converts solar radiation directly or indirectly into electrical energy by photoelectric effects. To make the solar power system provide enough power for the load, it is necessary to make a reasonable choice of solar panels according to the electrical power. The core lies in the determination of panel power and voltage which is chosen to be 20W,17.5V in this system. since the voltage of storage battery is 12V, the output voltage of the solar power should be larger and 17.5V is qualified. As to the choice of power, the specific calculation process is discussed below.

First, the amount of electricity consumed per day is supposed to be calculated. In the case where there is light in the daytime, the power consumption is composed by the load and battery. By using a USB power detector, the working voltage and working current of the load is measured to be 5.1V and 0.25A respectively, which indicates that the output power should be:

$$P_{load} = 5.1V \times 0.25A = 1.275W \tag{2-3}$$

according to the maximum charging voltage(15V) and charging current (2.1A) of the storage battery, its output power is：

$$P_{battery} = 15V \times 2.1A = 31.5W \tag{2-4}$$

If the weather station works 4 hours a day powered by the solar panel, the total power consumption is derived as:

$$W_{total} = (P_{\mathrm{load}} + P_{panel}) \times 4_h = 131.1\ (W \cdot h) \tag{2-5}$$

If the effective sunshine duration in Chengdu is 7 hours a day, considering the charging efficiency and the loss of the charging process, then the output power of solar panel would be

$$P_{panel} = W_{total} \div 7_h \div 85\% \approx 22W \tag{2-6}$$

In this equation, 85% is the estimated conversion efficiency of the controller.

2.4.3 Solar charge controller

The solar charge controller is an indispensable part of the entire PV power supply system. Due to the properties and limitations of the solar panel and light, the generated current is always a fluctuating curve. If the generated current is directly charged into the battery or supplied directly to the load, it is likely to cause damage to the battery and load, which would reduce their lifespan [18]. Therefore, we must first put the current into the solar charge controller. When the power is supplied to the load, the battery current is also transmitted first to the solar controller, and then the regulated current flows into the load. The purposes of this operation are: First, stabilize the discharge current. Second, ensure that the battery is not over discharge. Third, monitor and protect the load and the battery [19].

In addition to the basic control and protection, employed solar controller further includes a "maximum power point tracking" function. MPPT solar controller real-time detecting the solar panel voltage and current as well as constantly tracking the maximum power to keep the highest power generation efficiency of the entire system [20]. Furthermore, the indicator on the controller can visually display the battery power and whether the load and solar panel work, which facilities the work of system operators. This solar charge controller also provides 5V USB output port besides12V DC output, which means that the power is able to supply the entire device directly without voltage conversion. After the experimental measurement, the actual working voltage of the controller USB port is about 4.95V, which is slightly lower than 5.10V when the power is supplied by the computer, but its influence on the normal operation of the equipment is almost negligible.

2.4.4 Power management

Figure 14 shows the structure of power, Arduino mega2560 is driven by 5V USB port power supply, which comes from the solar power generation. Since all the modules applied in this station work in a voltage of 3.3V or 5V, they can be powered directly by the microcontroller.

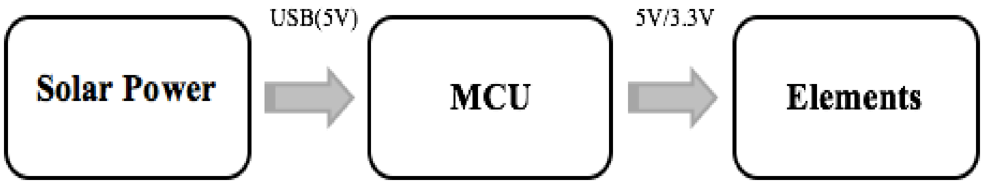


Figure 14. Flowchart illustrating the flow of electricity.

3 Problem encountered

This section discusses the problems I have met in this project and how I solved them to give a reference for readers who would like to do a similar device. Meanwhile, some existing limitations that are expected to be promoted are also provided.

3.1 Insufficient program space

I have tried other two types of microcontroller before Arduino mega2560. The initial attempt was mbed LPC1768 because it was the one I am familiar with, but I could not find a cloud server corporates with mbed and it was a hard work for me to construct a cloud by myself. SoI turned to Arduino Uno, a famous board for engineers, which has abundant accessible library as well as many existed cloud servers based on it. However, it turned out that 32KB program memory of Uno was not enough as 98% of the memory have been consumed when all the components were connected, which led to unstable performance of the device. That is the reason why I finally chose mega2560, compatible with the code of Uno, Arduino mega2560 only occupied 12% of the program space thanks to its 256KB flash memory.

3.2 Adjustment of pin numbers

The first choice of local display was another type called LCD1602, which asked for 6 digital pins of MCU. Though after replacing UNO by mega2560, the pin numbers are enough, it still caused a mass of wiring. Also, with the more sensors were added, the screen seemed too small to display all the information. These problems were solved when I found another similar display board with a larger screen, LCD12864, support two modes of connection and as mentioned in section 2.2.1, the SPI mode needs only 3 digital pins. [**Appendix 4**] shows how the new LCD reduces the number of connections.

3.3 Fail to obtain online timestamp

Though the final prototype is equipped with a real-time clock module to access the current time, my first attempt was to get the time from Internet. An important feature of network time is that all the weather stations in the network can access the same time and do not need the RTC setting on each unit. The core is to obtain the online timestamp and convert it into ordinary format, while it not possible to use a same Wi-Fi module to register cloud and the website of timestamp simultaneously. Fortunately, Qinglian Cloud is exploring a relative software interface to allow users catch timestamp by calling a specific call-back function.

3.4 A failed try of cloud server

It took me a long time to find a suitable and usable cloud sever. Jimupai is also an IOT cloud server in China but not immature enough. I have bought a Wi-Fi module produced by this company to public data. It worked when connecting the module to computer and using the serial port tool to send some specific command. However, it was hard for me to construct correct communication between the module and MCU. Qinglian cloud solved this problem by providing all the required SDK, so I do not have to consider the internal relationship but focus on realizing the function.

3.5 Software control of LCD brightness

As mentioned in section 2.2.1, the brightness of LCD is decided by whether the PSB pin (Figure 8) is powered or not. However, the initial idea is to power the module by digital output so that the operation of LCD is able to be controlled by the digital output signal with a button, but this method led to the instability because the display module could not be driven by a digital I/O pin. The improvement of the plan was to control PSB pin of LCD with a digital signal, previous issue had been solved but new problem appeared: the program of button status checking cannot be executed all the time. One case is that when the button is pressed, the MCU is running another program, resulting in no response of this action. Adding an interrupt may solve it but I had insufficient time remaining on the project to implement the routine properly using interrupts. Form another perspective, I eventually found the method of hardware control, which really simplified the problem.

3.6 Inaccurate timing

To upload data to cloud in every 10 seconds, a timer is employed to timing precisely, whereas the period of collecting dada is finished by built-in delay function, which works by pausing the program for the amount of time. The influence in every loop is small with this delay function, however, the period of storing information in external memory decided by 6 loops would be deviated a little from the desired 1 minute.

3.7 Hardware integration

The stability problem caused by breadboard wiring aroused when the initial prototype is completed. Disordered wiring will hinder the investigation of equipment failure and lead to their issues such as poor contact or short-circuit. Drawing a PCB board may be a solution but it is difficult to solder the Arduino board. Finally, I used an Arduino mega sensor shield to plug on the microcontroller, making it easier to integrate all the sensors and modules. Unfortunately, this shield board has only one I2C interface which is not possible to support the connection of light intensity sensor and real-time clock module simultaneously. The pin mapping of this sensor shield can be seen in [**Appendix 5**]. Considering that the performance of data collection can be evaluated from other 3 parameters whereas the real-time recording is an indispensable part of weather station, it is decided that the light is not measured in the final testing, but it should be noted that the code is still applicable to the case of light measurement. For modules that are not directly connected to the shield board, such as LCD and dust sensor, they are soldered to the hole board and then connected to the shield with wire as shown in Figure 15.

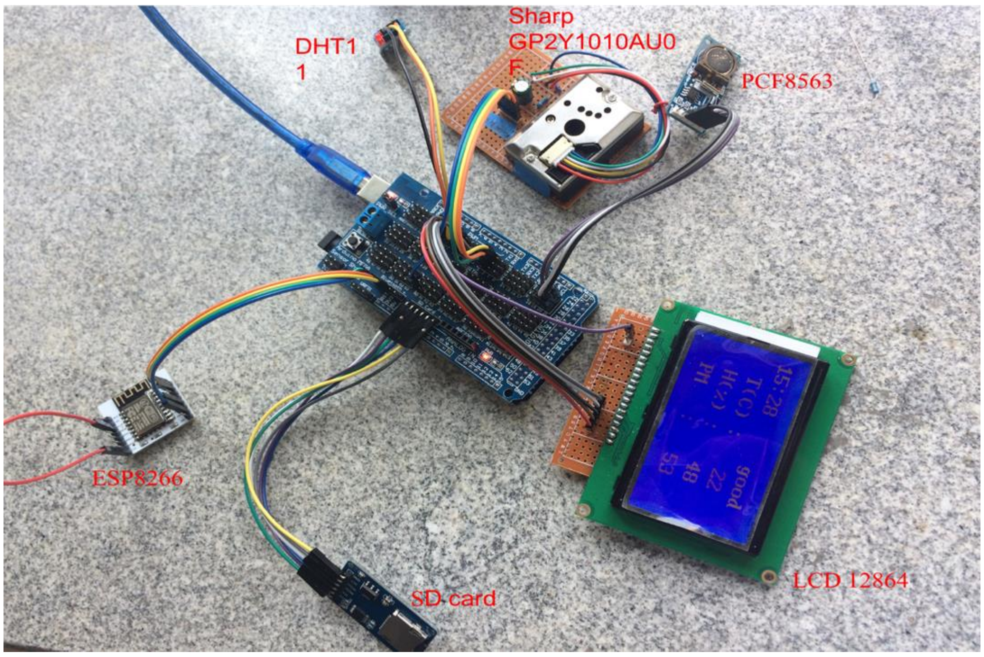


Figure 15. Hardware connection with sensor shield

4 Results and discussions

The complete unit could finally measure the value of temperature, relative humidity and dust density and defect their changes both in cloud and a local storage card, but it has to be placed indoor because there was no case to protect it from the damage of rains. Compared to a mechanical Thermo-Hygrometer, the deviation of temperature and humidity were around ± 2℃ and $\pm$ 4%. The dust density was not accurate enough as it was an estimated value, but the pollution status was same as that published by Chengdu Meteorological Bureau.

In order to evaluate the performance of the whole device, a testing was implemented. Figure 16 presents that the device was installed on the window platform of a toilet in UESTC campus with solar energy power supply. The information about this testing is listed in Table 4.

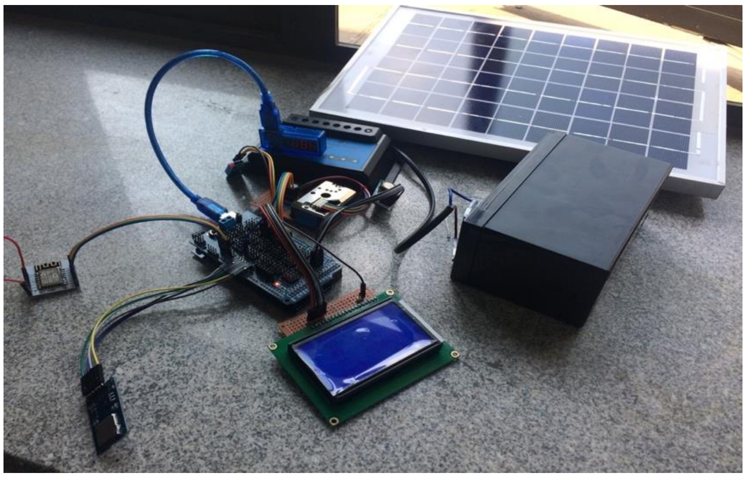


Figure 16. The complete prototype is placed on the window of toilet, powered by solar energy.

Table 4. The details of test condition

Date	Storage battery	Weather condition	Performance
15/4/2017(0:00-24:00)	Full charged	Sunny	Every part is working

After one-day operation, we found that the storage battery was still full charged, which meant that the solar panel had generated power during the daytime. Moreover, we could compare the data uploaded to cloud and data stored in SD card in Figure 17 and Figure 18. Although the cloud collected information every 10 seconds and SD card stored the average value of 6 samples once a minute. It can be seen that the line graph plotted in excel based on local storage data is close to that displayed in cloud server. These figures also show that the temperature reached the maximum at two in the afternoon and at this time the relative humidity is the lowest, which is fully consistent with the weather changes of a day. However, the dust density almost remains constant except for several peaks, because when the dust density is below a certain value, the device will set it to the minimum value directly, the specific reasons have been explained in section 2.1.3. The appearance of those peaks may be the result of smoking.

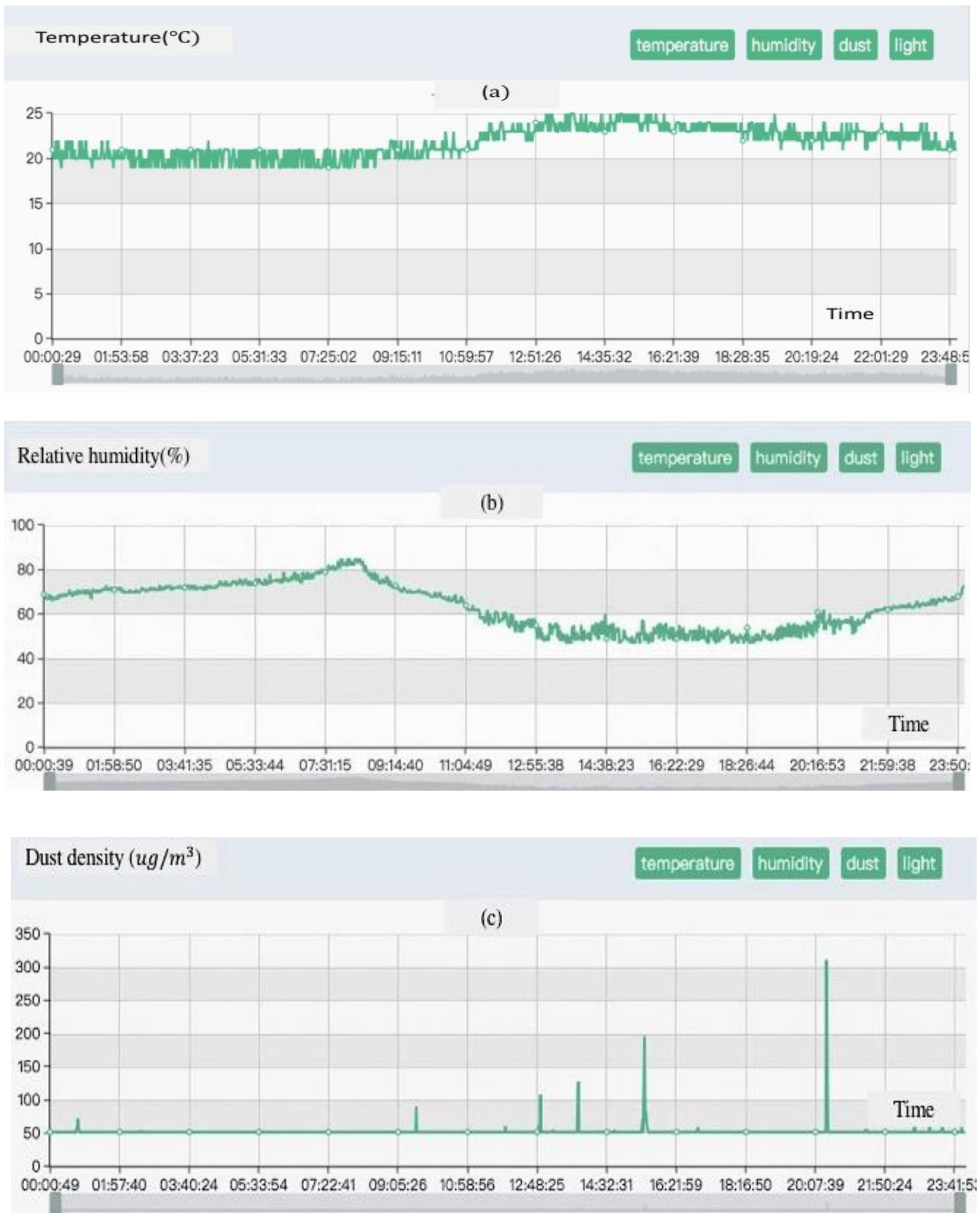


Figure 17.The cloud website showing the change of (a) temperature(b)relative humidity (c)dust density on 15th April

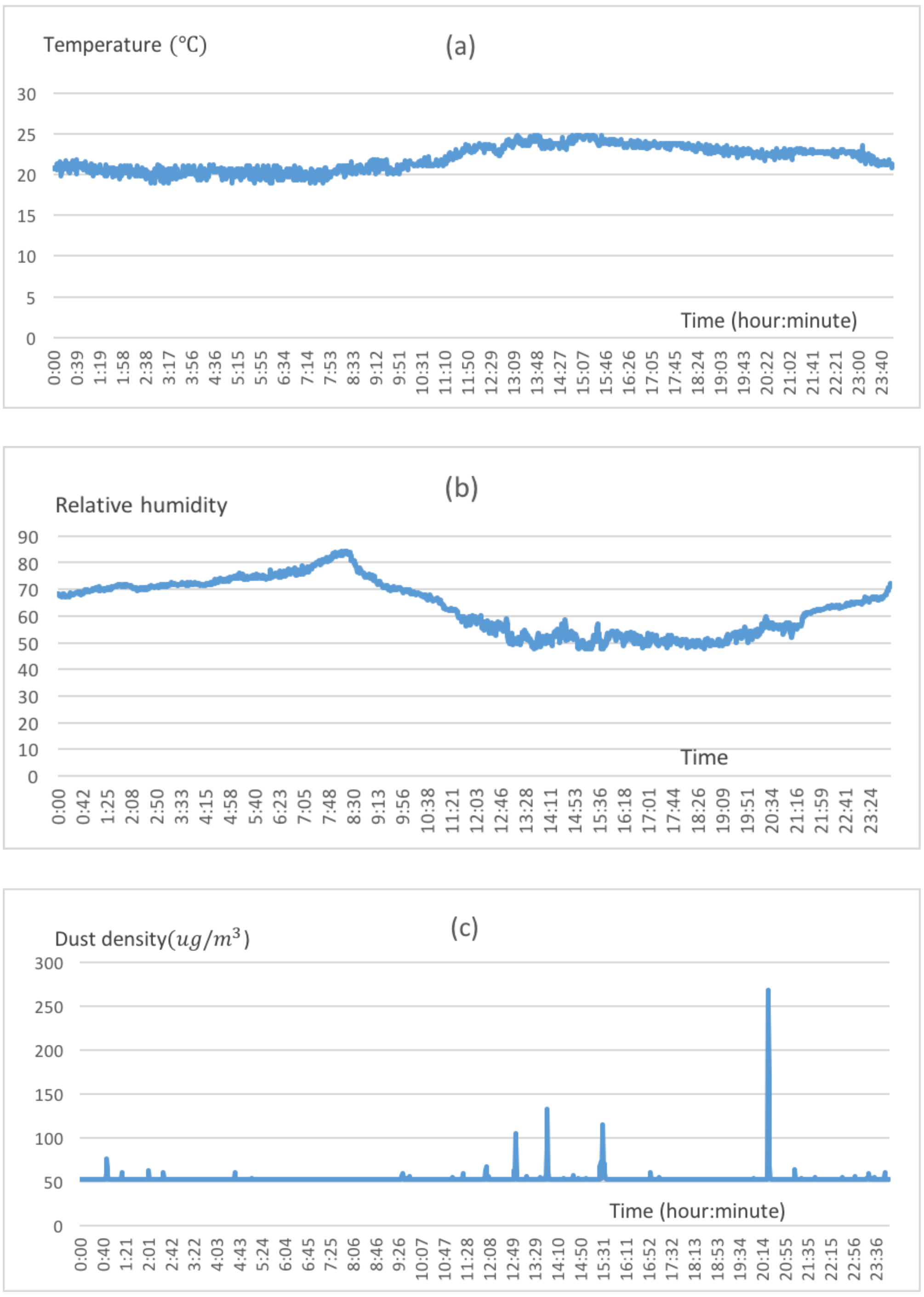


Figure 18. The plot based on the data stored in SD memory showing the change of(a) temperature (b)relative humidity (c)dust density on 15th April.

5 Conclusion and future work

A cloud based, cost effective micro-weather station is proven to be realisable throughout this report. Except for a good performance in gathering climate information, it has additional functions of local display, local storage and cloud observation. This device can be applied for home users to measure small-range environmental conditions automatically supplied by solar power but not very suitable for precisely detection since the sensors are not accurate enough. Future work may focus on following aspects:

Firstly, add sensors like wind speed and wind direction to acquire more meteorological information. Attention should be paid to the power management as these sensors are driven by 12V-24V power supply, that is, they should be powered by the storage battery directly other than MCU.

Then, design a suitable enclosure capable of withstanding the outdoor environment and operating for at least 1 year without maintenance. The shell is expected to ventilate well so that the temperature sensor and dust density sensor could collect more correct measurements. Also, there should be an open window on the case to permit the LCD display.

Next, a function of visiting local memory from the device is expected to add. This may help to observe and delete historic recordings in the locality. As the increase of modules, the storage room is likely to get full, if such problem occurs, the device is supposed to store the new information by recovering the old one.

Eventually, it would be a meaningful experimentation to access several weather stations in Qinglian cloud and evaluate their performance.

Reference

[1] H. P. Yoke, "An Introduction to Science and Civilization in China," Mineola: Dover Publications. p.105.

[2] M. Benghanem, "Measurement of meteorological data based on wireless data acquisition system monitoring," *Applied Energy*, 2009, pp. 2651–2660.

[3] A. Kadri, E. Yaacoub, M. Mushtaha and A. Abu-Dayya, "Wireless sensor network for real-time air pollution monitoring," *2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, Sharjah, 2013, pp. 1-5.

[4] E. Kanagaraj, L. M. Kamarudin, A. Zakaria, R. Gunasagaran and A. Y. M. Shakaff, "Cloud-based remote environmental monitoring system with distributed WSN weather stations," *2015 IEEE SENSORS*, Busan, 2015, pp. 1-4.

[5] S. Rosiek, F.J. Batlles, " A microcontroller-based data-acquisition system for meteorological station monitoring," *49th Energy Conversion and Management*, 2009, pp.3746-3754.

[6] A. Shaout, Yulong Li, M. Zhou and S. Awad, "Low cost embedded weather station with intelligent system," *2014 10th International Computer Engineering Conference (ICENCO)*, Giza, 2014, pp. 100-106.

[7] Arduino Mega 2560, Available: https://www.arduino.cc/en/Main/ArduinoBoardMega

[8] X. J. Zhou, X. H. Su, Chi, M. Y. Yuan, et al, "Forecast of air pollution index based on BP neural network," *Journal of Harbin Institute of Technology*, 2004, 36(5), pp. 582-585.

[9] H. M. Bai, R. P. Shen et al, "Forecasting Model of Air Pollution Index Based on BP Neural Network," *Environmental Science & Technology*, 2013, 36(3), pp. 186-189.

[10] [Technical Regulation on Ambient Air Quality Index, 2012, Ministry of Environmental Protection of the People’s Republic of China, pp:2-4.

[11] *Application note of Sharp dust sensor GP2Y1010AU0F*, SHARP,2006, pp.1-9.

[12] *TSL2560, TSL2561 LIGHT-TO-DIGITAL CONVERTER*, Texas Advanced Optoelectronic Solutions Inc,2007, pp.1-38.

[13] *PCF8563 Real-time clock/calendar Product data sheet*, NXP Semiconductors, 2008, pp.1–32.

[14] A. Tapashetti, D. Vegiraju and T. Ogunfunmi, "IoT-enabled air quality monitoring device: A low cost smart health solution," 2016 IEEE Global Humanitarian Technology Conference (GHTC), Seattle, WA, 2016, pp. 682-685.

[15] J. Han, J. Lee, E. Lee and D. S. Kim, "Development of a low-cost indoor environment monitoring system based on a hybrid wireless sensor network," *2016 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, 2016, pp. 461-462.

[16] G. C. Fox, S. Kamburugamuve and R. D. Hartman, "Architecture and measured characteristics of a cloud based internet of things," *2012 International Conference on Collaboration Technologies and Systems (CTS)*, Denver, CO, 2012, pp. 6-12.

[17] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol, Version 1.2",2002.

[18] M. Hossain, "A review of principle and sun-tracking methods for maximizing". *Renewable and Sustainable Energy Reviews 13*, 2009, pp.1800–1818.

[19] E.V. Dyketal. "Long-term monitoring of photovoltaic devices, " *Renew Energy* . 2002.

[20] E Koutroulis, K. Kalaitzakis, “Development of a microcontroller based photovoltaic maximum power point tracking control system,” *IEEE Transactions on Power Electronics*. 2001

Appendix 1

Main code programmed by Arduino IDE

```
/************* Library source***************** *************/
#include <SD.h>
#include <MsTimer2.h>
#include <DHT.h>
#include <ql_interface.h>
#include <Wire.h>
#include "TSL2561.h"
#include <LCD12864RSPI.h>
#include "GP2Y1010AU0F.h"
#include <Rtc_Pcf8563.h>
Rtc_Pcf8563 rtc;
double data[4];                          //store 4 types of climate data
double sum[4];                           //store the sum of 4 types of climate data
int count = 0;                           //record the times
/************* WIFI define ***************** *************/
#define PRODUCT_ID  "1000129322"                  //product ID
#define PRODUCT_KEY "3474667248552863269a22af6d36649b"//product code
#define PRODUCT_VER "01.01"                       //product version
QLCloud    qlcloud;
bool is_wifi_init = false;
/***********************************************************/

/************* SD card define ******************************/
const int chipSelect = 53;
/***********************************************************/

/************* LCD define **********************************/
#define AR_SIZE( a ) sizeof( a ) / sizeof( a[0] )
unsigned char show1[] = "weather station";
unsigned char show2[] = "Yan Danli";
/***********************************************************/

/************* DHT11 define ********************************/
#define DHTPIN       14
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
/***********************************************************/

/************* dust sensor define **************************/
#define PIN_LED 30
#define PIN_OUTPUT A0
GP2Y1010AU0F GP2Y1010AU0F(PIN_LED, PIN_OUTPUT);
/***********************************************************/

/************* light sensor define *************************/
```

```
TSL2561 tsl(TSL2561_ADDR_FLOAT);
/**************************************************************/

int32_t SerialSendBytes(uint8_t*data, int32_t len)
{
  Serial.write(data, len);
}
/*initialize the Wi-Fi module*/
void init_wifi()
{
  qlcloud.init((uint8_t*)PRODUCT_ID,                    (uint8_t*)PRODUCT_KEY,
(uint8_t*)PRODUCT_VER);
}
void upload_data(char *key, double weather) {
  int ret = 0;
  char value[16];
  itoa(weather, value, 10);
  qlcloud.upload_data((uint8_t*)key, strlen(key), (uint8_t*)value, strlen(value));
}

/*query the status*/
void query_satus_cb(int32_t status)
{
  if (status == STATUS_NOT_INIT) {           //initialization finish
    is_wifi_init = false;
    init_wifi();
  } else {                                  //initialization unfinish
    is_wifi_init = true;
  }
}

/*timer*/
void timer_handle()
{
  static int number = 0;

  if (!is_wifi_init) {                      //wifi uninitialized
    qlcloud.query_module_status();
  }
  else {
    switch (number) {                       //send climate data every 10s
      case 0:
        upload_data("temperature", data[0]);
        number = 1;
        break;
      case 1:
        upload_data("humidity", data[1]);
        number = 2;
        break;
      case 2:
        upload_data("dust", data[2]);
        number = 3;
        break;
      case 3:
        upload_data("light", data[3]);
        number = 0;
        break;
    }
  }
}
/*data aquisition function*/
void collect_data() {
  data[0] = (int)dht.readTemperature();                    //read temperature

  data[1] = (int)dht.readHumidity();                       //read humidity

  data[2] = (double)  GP2Y1010AU0F.getDustDensity();     //read dusy density

  uint32_t lum = tsl.getFullLuminosity();
  uint16_t ir, full;
  ir = lum >> 16;
  full = lum & 0xFFFF;
  data[3] = (double)tsl.calculateLux(full, ir);                 //read light Luminosity
}
```

/***calculate average function***/

```
void cal_average() {
  for (int i = 0; i < 4; i++) {                             //process each data from data[0] to data[3]
    sum[i] = sum[i] + data[i];                              //get the sum of 6 samples
    if (count >= 6) {
      data[i] = sum[i] / 6;                               //calculate the average value
      sum[i] = 0;                                       //reset sum
    }
  }
}
```

/*sd card storage function*/

```
void sd_storage() {

  String dataString = "";                                  //define an arrary to store imformation
  for (int i = 0; i < 4; i++) {
    dataString += String(data[i]);
    dataString += ",";
  }                                                   //write 4 types of climate data into the array
  dataString += String(rtc.formatDate());                  //write date into the array
  dataString += ",";
  dataString += String(rtc.formatTime(2));                 //write time into the array
  File dataFile = SD.open("datalog.txt", FILE_WRITE);           //open "datalog.txt"writing
configuration"
  if (dataFile) {
    dataFile.println(dataString);                         //send datastring to the file
    dataFile.close();                                     //close the file
  }
}
```

/*LCD display function*/

```
void lcd_display() {

  LCDA.CLEAR();                                           //clear the screen
  LCDA.DisplayString(0, 0, (unsigned char *)rtc.formatTime(2), 3 *
AR_SIZE(rtc.formatTime(2)));

  int grade = GP2Y1010AU0F.getGradeInfo(data[2]);       //display the pollution status
  char *grade_show[] = {"perfect", "good", "light", "medium", "heavy", "severe"};
  LCDA.DisplayString(0, 5, (unsigned char *)grade_show[grade], 3 *
AR_SIZE(grade_show[grade]));

  char *name_show[] = {"T(C) :", "H(%) :", "PM   :"};
  char data_show[3];
  for (int j = 0; j < 3; j++) {
    dtostrf(data[j], 3, 0, data_show);
    LCDA.DisplayString(j + 1, 1, (unsigned char *)name_show[j], 3 *
AR_SIZE(name_show[j]));
    LCDA.DisplayString(j + 1, 5, (unsigned char *)data_show, AR_SIZE(data_show)); }}

void setup()
{

  Serial.begin(115200);
  qlcloud.reg_send_bytes_cb(SerialSendBytes);      //register the callback function of sending
serial bytes
  qlcloud.reg_query_status_cb(query_satus_cb);    //register the callback function of querying
status
  qlcloud.set_wifi_debug_status(1);
  dht.begin();                                     //initialize temp/humi seneor
  tsl.begin();                                     //initialize light sensor
  tsl.setGain(TSL2561_GAIN_16X);
  tsl.setTiming(TSL2561_INTEGRATIONTIME_13MS);
  SD.begin(chipSelect);                            //initialize SD card module
  LCDA.Initialise();                               //initialize LCD display
  LCDA.DisplayString(1, 0, show1, AR_SIZE(show1));
  LCDA.DisplayString(3, 2, show2, AR_SIZE(show2));
  MsTimer2::set(10000, timer_handle);          //set a timer to upload data every 10 seconds
  MsTimer2::start();
}
```

```
void loop()
{
  count++;
  collect_data();                          // collect all the climate data
  lcd_display();                           //display real-time data
  cal_average();                           //calculate the average value of each data
  if (count >= 6) {                        //transsmit average data to sd storage
    sd_storage();
    count = 0;
  }
  int available_num = Serial.available();
  while (available_num--) {
    qlcloud.rx_byte(Serial.read());
  }                                   //read serial port buffer

  qlcloud.rx_data_handler();                 //handle serial port data
  delay(10000);

}
```

Appendix 2

Individual Air Quality Index(IAQI) and corresponding pollutant concentration limit

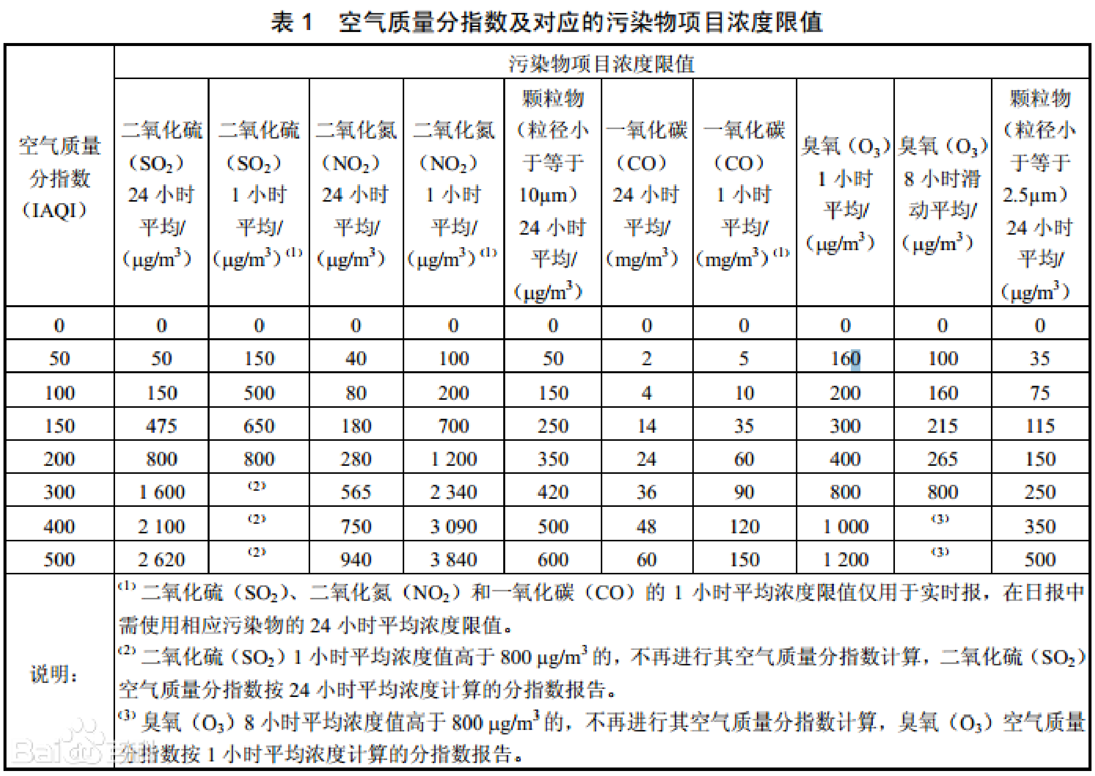


表 1　空气质量分指数及对应的污染物项目浓度限值

空气质量分指数（IAQI）	污染物项目浓度限值									
	二氧化硫（SO_2）24 小时平均/（μg/m^3）	二氧化硫（SO_2）1 小时平均/（μg/m^3）[1]	二氧化氮（NO_2）24 小时平均/（μg/m^3）	二氧化氮（NO_2）1 小时平均/（μg/m^3）[1]	颗粒物（粒径小于等于 10μm）24 小时平均/（μg/m^3）	一氧化碳（CO）24 小时平均/（mg/m^3）	一氧化碳（CO）1 小时平均/（mg/m^3）[1]	臭氧（O_3）1 小时平均/（μg/m^3）	臭氧（O_3）8 小时滑动平均/（μg/m^3）	颗粒物（粒径小于等于 2.5μm）24 小时平均/（μg/m^3）
0	0	0	0	0	0	0	0	0	0	0
50	50	150	40	100	50	2	5	160	100	35
100	150	500	80	200	150	4	10	200	160	75
150	475	650	180	700	250	14	35	300	215	115
200	800	800	280	1 200	350	24	60	400	265	150
300	1 600	[2]	565	2 340	420	36	90	800	800	250
400	2 100	[2]	750	3 090	500	48	120	1 000	[3]	350
500	2 620	[2]	940	3 840	600	60	150	1 200	[3]	500
说明：	[1] 二氧化硫（SO_2）、二氧化氮（NO_2）和一氧化碳（CO）的 1 小时平均浓度限值仅用于实时报，在日报中需使用相应污染物的 24 小时平均浓度限值。 [2] 二氧化硫（SO_2）1 小时平均浓度值高于 800 μg/m^3 的，不再进行其空气质量分指数计算，二氧化硫（SO_2）空气质量分指数按 24 小时平均浓度计算的分指数报告。 [3] 臭氧（O_3）8 小时平均浓度值高于 800 μg/m^3 的，不再进行其空气质量分指数计算，臭氧（O_3）空气质量分指数按 1 小时平均浓度计算的分指数报告。									

Appendix 3

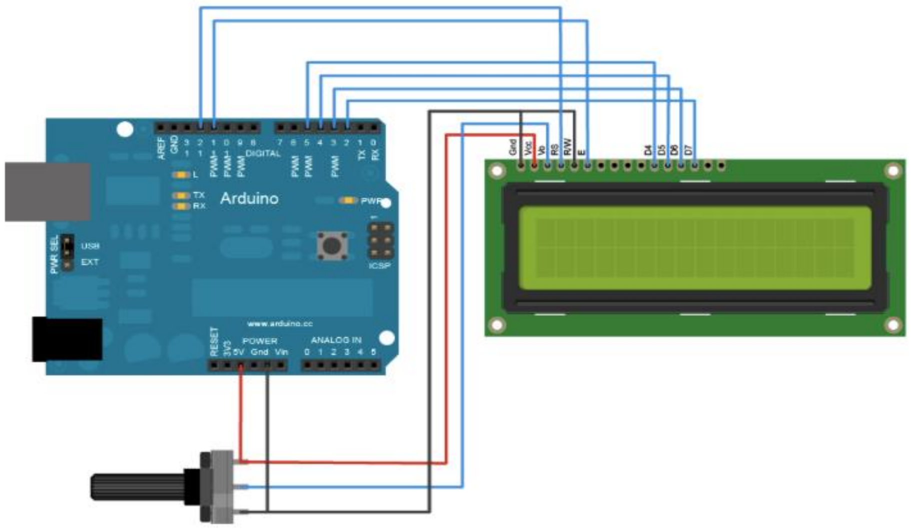
The hourly data stored in micro-SD card.

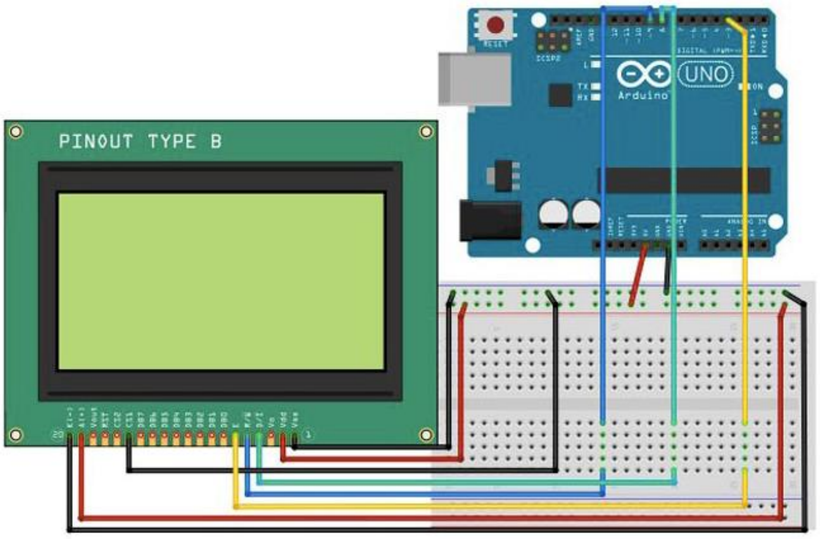
```
20.83,68.33,53.00,703.00,04/15/2017,00:00
20.67,68.33,53.00,703.00,04/15/2017,00:01
20.67,67.83,53.00,703.00,04/15/2017,00:02
20.67,67.50,53.00,703.00,04/15/2017,00:03
20.67,67.83,53.00,703.00,04/15/2017,00:04
21.33,67.50,53.00,703.00,04/15/2017,00:05
20.50,67.33,53.00,703.00,04/15/2017,00:06
20.83,67.83,53.00,703.00,04/15/2017,00:07
21.33,68.50,53.00,703.00,04/15/2017,00:08
21.17,68.17,53.00,703.00,04/15/2017,00:09
21.50,68.17,53.00,703.00,04/15/2017,00:10
21.33,67.50,53.00,703.00,04/15/2017,00:11
19.83,67.00,53.00,703.00,04/15/2017,00:12
20.33,67.00,53.00,703.00,04/15/2017,00:13
21.00,67.67,53.00,703.00,04/15/2017,00:14
20.50,67.50,53.00,703.00,04/15/2017,00:15
20.67,67.67,53.00,703.00,04/15/2017,00:16
21.33,67.67,53.00,703.00,04/15/2017,00:18
20.67,67.33,53.00,703.00,04/15/2017,00:19
20.83,67.50,53.00,703.00,04/15/2017,00:20
21.67,68.67,53.00,703.00,04/15/2017,00:21
20.83,67.83,53.00,703.00,04/15/2017,00:22
21.00,67.83,53.00,703.00,04/15/2017,00:23
21.00,68.17,53.00,703.00,04/15/2017,00:24
20.33,67.83,53.00,703.00,04/15/2017,00:25
21.33,68.50,53.00,703.00,04/15/2017,00:26
20.67,68.00,53.00,703.00,04/15/2017,00:27
20.17,68.17,53.00,703.00,04/15/2017,00:28
20.83,68.33,53.00,703.00,04/15/2017,00:29
21.00,68.67,53.00,703.00,04/15/2017,00:30
21.17,68.83,53.00,703.00,04/15/2017,00:31
21.67,68.67,53.00,703.00,04/15/2017,00:32
21.33,69.00,53.00,703.00,04/15/2017,00:33
20.67,68.67,53.00,703.00,04/15/2017,00:34
20.50,68.33,53.00,703.00,04/15/2017,00:35
21.50,68.67,53.00,703.00,04/15/2017,00:36
20.83,68.50,53.00,703.00,04/15/2017,00:37
21.83,69.00,53.00,703.00,04/15/2017,00:38
20.00,68.00,53.00,703.00,04/15/2017,00:39
21.33,68.17,53.00,703.00,04/15/2017,00:40
21.17,68.83,53.00,703.00,04/15/2017,00:41
21.00,68.33,53.00,703.00,04/15/2017,00:42
21.00,68.50,53.00,703.00,04/15/2017,00:43
21.33,68.83,53.00,703.00,04/15/2017,00:44
21.33,69.33,59.14,703.00,04/15/2017,00:46
21.17,69.17,75.42,703.00,04/15/2017,00:47
20.83,69.83,64.63,703.00,04/15/2017,00:48
21.33,69.67,57.29,703.00,04/15/2017,00:49
21.00,69.83,53.00,703.00,04/15/2017,00:50
20.50,69.00,53.00,703.00,04/15/2017,00:51
20.83,69.50,53.00,703.00,04/15/2017,00:52
```

```
21.00,69.33,53.00,703.00,04/15/2017,00:53
21.50,69.50,53.00,703.00,04/15/2017,00:54
20.83,69.17,53.00,703.00,04/15/2017,00:55
20.67,69.33,53.00,703.00,04/15/2017,00:56
20.67,69.50,53.00,703.00,04/15/2017,00:57
20.33,69.50,53.00,703.00,04/15/2017,00:58
20.50,69.33,53.00,703.00,04/15/2017,00:59
```

Appendix 4

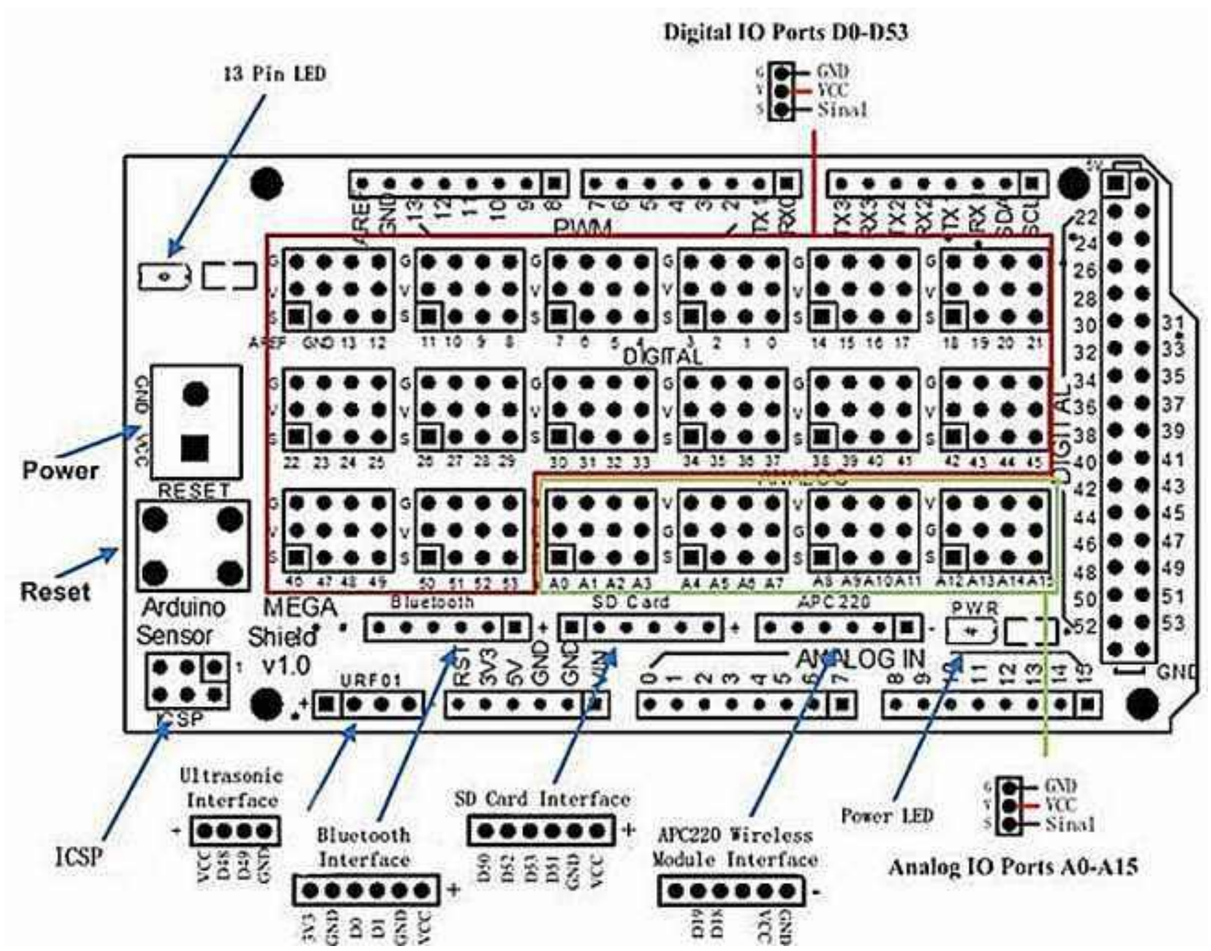
The upper figure shows the hardware connection between Arduino and LCD1602, 6 digit Pins are required. The next figure shows that the digit pins are reduced to 3 by using SPI mode of LCD12864.





Appendix 5

The pin mapping of Arduino Mega2560 sensor shield


Digital IO Ports D0-D53
GND
VCC
Sinal
13 Pin LED
Power
Reset
ICSP
Ultrasonic Interface
VCC D18 D19 GND
Bluetooth Interface
3V3 GND D0 D1 GND VCC
SD Card Interface
D50 D52 D53 D51 GND VCC
APC220 Wireless Module Interface
D19 D18 VCC GND
Power LED
Analog IO Ports A0-A15
GND
VCC
Sinal
Arduino Sensor
MEGA Shield v1.0
DIGITAL
ANALOG IN
PWM