

Design, Implementation and Evaluation of Remote Weather Monitoring System

Abdulkadir H. Alkali¹, Stephen J. Bassi², Emmanuel G. Dada³, Zarah M. Bukar⁴,
Mohammed Idris⁵, Muhammad H. Abdulfattah⁶

^{1,2,3,4,5,6}Department of Computer Engineering, University of Maiduguri, Maiduguri, Nigeria
(¹ahalkali@unimaid.edu.ng, ²sjbassi74@unimaid.edu.ng, ³gbengadada@unimaid.edu.ng)

Abstract-The measurement of weather variables today is of great importance to both farmers and researchers. These investigations reveal several indicators that contribute to inventive decision making in research and farming. The conventional manual techniques for weather monitoring are prone to errors and can be both time consuming and tedious. This paper presents an automated weather monitoring system. The system was designed, implemented and evaluated using electronic sensors. The proposed system can monitor and measure temperature, humidity, wind speed, wind direction, solar illumination and rainfall at near real-time. Remote monitoring was made possible by XBee modules at the transmitting and receiving units. Upon receiving weather parameters, a data logger saves the values on a PC. The system was evaluated using existing monitors. Correlation of 75%, 99% and 99% were obtained for the wind speed, humidity and temperature variables between the developed and existing systems. A 100% resemblance between the developed and existing system was obtained for the wind direction monitoring. Rainfall was simulated using a syringe of known volume and the system was able to record it with 96.8% accuracy.

Keywords- Remote Monitoring, Data Logger, Sensor, Weather

I. INTRODUCTION

Weather is the state of the atmosphere, to the degree that it is hot or cold, wet or dry, calm or stormy, clear or cloudy [1]. Meteorological parameters such as temperature, humidity and wind speed among others, can be monitored and measured using weather stations. These parameters for an area can be measured by a weather station [2] through manual or automated techniques. Although meteorological satellites can be an alternative system, conventional weather stations are preferred due to their better accuracy even at higher cost [3]. Conventional weather station may involve manual or automated monitoring and can also be used to predict future weather conditions of the area. In manual monitoring, consistent human interaction with measuring equipment such as thermometer for measuring temperature and rain gauge for rainfall at regular interval over a long period of time. Due to

the time involved and the consistency, manual weather measurement is difficult. This is as a result of human nature such as fatigue, forgetfulness, errors and accessibility especially in hard to reach areas.

Automated monitoring on the other hand involve the use of sensors such as temperature and humidity sensors attached to microcontrollers to measure and either store or transmit the parameters without human interaction. Their main issue is the power to keep them running over long periods.

The weather station is mostly placed at a remote location and thus the monitored parameters have to be manually accessed at periodic intervals or automatically transmitted to a secure location. With recent advancements, weather stations are mostly equipped to the capability of automatic transmission of monitored parameters to a data logger for evaluation and storage. Development boards such as Arduino are inexpensive, open source, easy to set-up and availability of many sensors that are easily interfaced with [4] has made them a popular choice in embedded system designs.

Several works [1, 2, 4 – 8] have proposed different weather monitoring systems. Some of these systems are able to monitor few weather parameters while those that measured more parameters used light dependent resistors as light intensity sensor which gives inaccurate measurements. In addition, these systems were not compared with existing weather measuring systems to validate their efficacies.

In this work, a weather station to monitor Temperature, Humidity, Sun intensity, Rainfall, Wind speed and direction was designed, implemented and evaluated using existing systems. To achieve the implementation, sensors to measure temperature, humidity, wind speed, wind direction, rainfall and light intensity were used. The sensors were interfaced to an Arduino board which also served as the processing unit. XBee modules were used to transmit and receive the sensors' data which were saved to a database via a data logger. To enhance the reliability of the system and allow it to run independently and at a remote location, a solar based power supply was used. The power supply charged a battery that both supplied the required power to the system and saved some for use during no-sun periods.

II. REVIEW OF PREVIOUS WORK

The effect of climate change on life is no longer news as it continues to negatively affect the quality of life. Monitoring the weather is therefore a crucial factor in observing daily effects of climate change and also towards limiting it. In order to achieve effective weather monitoring, various works towards achieving automated measurements of weather parameters through weather stations have been proposed. Some related works include:

An Arduino based weather monitoring system was developed by [1] to monitor temperature, humidity and light intensity. The measured parameters were fed to a PC and saved as a text file that can be accessed for future reference and also simultaneously displayed on an LCD screen. The temperature was an average of the measurements obtained from a DHT 11 and LM35 sensors. Humidity and light intensity were measured using DHT 11 and a light dependent resistor (LDR) respectively. The system could only monitor three parameters and using an LDR as a sensor for measuring sunlight intensity does not give an accurate reading.

In [2], a weather monitoring system that consisted of a temperature and humidity sensor was developed. The system was developed using Arduino Uno board and a DHT 11 sensor to monitor the temperature and humidity in a room and save it in MS Excel. The system could only measure temperature and humidity.

The Arduino-Based Weather Station reported in [4] measured temperature, relative humidity and barometric pressure. A solar-based power system was used to recharge a battery for powering the weather station. The measured parameters were saved in a micro SD card and also wirelessly transmitted to a remote receiver. The system measured only three weather parameters.

In [5], a wireless sensor network to measure temperature, relative humidity, atmospheric temperature, illuminance, UV light intensity, Carbon dioxide and Volatile organic components was developed. The WSN consisted of the sensor node, gateway and server. The sensor node acquired sensor data and sends same to the server through the gateway while the gateway routed the data to the server using an Ethernet connection and the server collects data from the individual sensors and stores them in a database. The system is not a complete weather monitoring system as it did not measure rainfall, wind speed and wind direction.

Weather parameters were measured using an Arduino and various sensors in [6]. The parameters measured were temperature, humidity, barometric pressure and rain drop sensor. The measured parameters were transmitted to a website using a WI-FI shield. Only three weather parameters were measured while the rain drop sensor does not measure rainfall, rather indicates the presence or absence of rainfall.

In the work reported in [7], the weather station consisted of an LM35, anemometer, wind vane and LDR interfaced to a microcontroller to measure temperature, wind speed, wind

direction and sunlight intensity respectively. The measured data was displayed on an LCD attached to the station and also sent to a user via short message service (SMS). The system used LM35 for temperature monitoring which is not weatherproof. Using an LDR to directly measure intensity does not give reliable data while SMS service could be expensive over time.

Susmitha and Sowmyabala [8] designed an embedded weather monitoring system for monitoring weather parameters in an industry. The system used temperature, gas, humidity and accelerometer sensors to measure temperature, combustible gas presence, humidity and earthquake condition respectively. The measured parameters were transmitted to a mobile device via SMS and also saved on PC. Only temperature and humidity are weather parameters and using SMS service makes the system expensive.

The contribution of this work is the measurement of various weather parameters as against selected few. The measured parameters such as temperature, humidity, wind speed, wind direction and rainfall were evaluated against existing systems to ascertain its reliability. Another important contribution is the use of solar power to run the system and also constantly monitor its level of charge remotely just as the weather parameters.

III. MATERIALS AND METHODS

The procedure taken toward the design, implementation and evaluation of the remote weather station was described in this section. Sketches were written as algorithm to perform the software interfacing and hence read parameters from each of the devices and then convert them to appropriate values. Within the algorithm, calculations were also included such as conversion of units. Decisions were also made such as whether to read the variables and transmit or to idle.

On the output side, XBee module and an LCD were interfaced to transmit recorded data with time stamp to a remote location (another ZigBee) and to display the values respectively. Figure 1 shows the flowchart of the weather station.

The block diagram of the proposed system consisting of a transmitter and receiver sections are shown in figure 2 and figure 3 respectively. The transmitter section served as the weather monitoring unit and consists of the various parameter sensors for measuring; temperature and humidity, illumination, wind speed, wind direction and rainfall. The solar based power supply is also found at the transmitter. A liquid crystal display (LCD) and an XBee module are also at the transmitter section to display and transmit measured parameters respectively. An Arduino board was used as the processing unit of the system.

The receiver section has an Arduino board, XBee module and the data logger as the processing unit, receiver and data logging unit respectively.

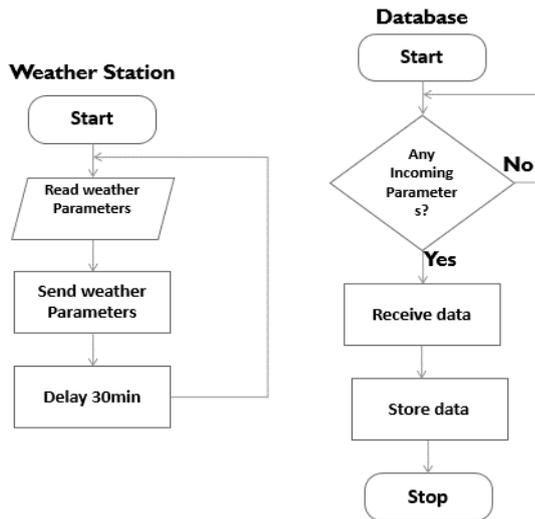


Figure 1. Flowchart of the Weather Station

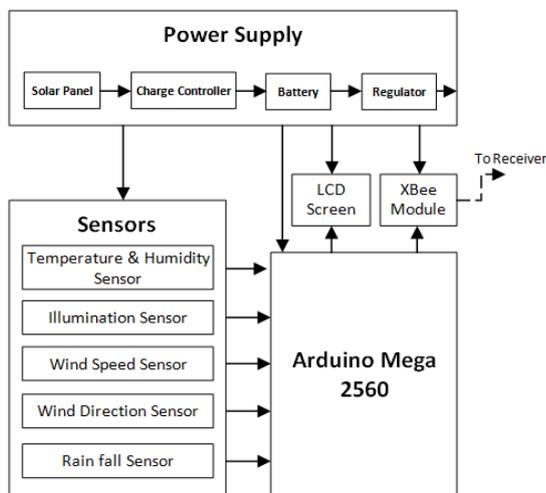


Figure 2. Block diagram of the weather station

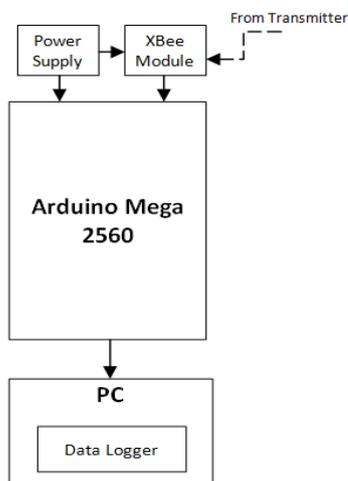


Figure 3. Block Diagram of the Base Station

A. Arduino Mega 2560

A microcontroller is a self-contained system with peripheral, memory and processor that can be used as an embedded system. Most programmable microcontrollers that are used are embedded in other consumer product or machinery including phone, peripherals, automobiles and household appliances for computer system. Some of these microcontrollers are more sophisticated while others have minimal requirement for memory and programming length and low software complexity. It also contained general purpose input/output pins of which it varies depending on the microcontroller.

An Arduino Mega2560 is used for this project. Arduino is an open source electronics prototyping platform based on flexible, easy to use hardware and software. All Arduino board are programmed through the Arduino IDE called sketch, it is an open source physical computing platform based on the simple microcontroller board and a development environment for writing to the board. Arduino can use interactive objects taking from a variety of sensors, and controlling a variety of actuators and other physical output. Table 1 tabulates major Arduino board specifications and served as a comparison chart for our choice.

TABLE I. ARDUINO BOARDS COMPARISON

Item	System voltage (V)	Clock speed (MHz)	Digital input/output	Analog input	PWM	Memory (KB)
Arduino Uno R3	5	16	14	6	6	32
Arduino Pro	3.3	8	14	6	6	32
Arduino Pro mini	3.3	8	14	8	6	32
Arduino Mega 2560	5	16	54	16	14	256
Arduino Mega Pro	3.3	8	54	16	14	256

The Arduino Mega 2560, shown in figure 4, has 10 bits analogue-to-digital converter, extra input/output pins, higher memory space and very high clock speed and was thus used for this work since many sensors were used, hence requiring more input/output ports. It also requires a 5 volt power supply which is easy to provide using a simple 5 volt regulator.

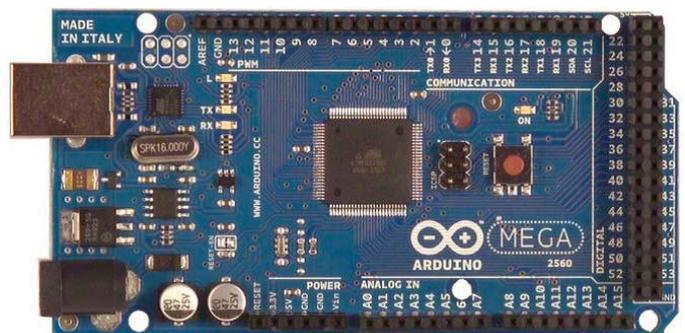


Figure 4. Arduino Mega 2560

From figure 2, the sensors were interfaced to the Arduino Board to read temperature, humidity, light intensity, wind speed, wind direction and rain as inputs. The sensors were connected to the Board as indicated in table 2.

TABLE II. PIN CONFIGURATION OF SENSORS ON ARDUINO

S/NO	SENSING DEVICES	PIN DECLARATION
1	DHT11	Digital pin 4
2	LDR	Analog pin A1
3	LCD	Digital pins 8,9,10,11,12,13
4	Rain gauge	Digital pin 2
5	Wind Speed	Digital pin 3
6	Wind direction	Analog pin A0
6	XBee	Digital PIN Rx0,Tx1
7	LED	Digital pins 6,7
8	Battery Level Sensor	Analog pin A2

B. Power supply

As the system requires a very low power we do not give extra emphasis on the power supply system. We used a 20w solar panel to charge a 12V, 7AH deep cycle battery. The solar panel was used as a source of independent power supply since the weather station was to be used for remote sensing. Mains power supply is unreliable while fuel powered alternative is expensive and requires constant human intervention through monitoring. A charge controller was used for charging the battery at 14 volts. From the battery, a cigar adapter charger with an output of 5.4 – 5.5 volts at 1.5 A was used. The charger has an inbuilt converter circuit to remove any ripple or electromagnetic interference that may occur. Being solar charges, high spikes are not expected as in the case of vehicle charged batteries. In any case, the the cigar adapter charger was equipped to handle such.

A 20 watt solar panel can generate (over 7 hours) daily:

$$Power\ generated\ (Pg) = 20 \times 7 = 140Wh$$

For the chosen battery, the total power is:

$$Battery\ power\ (Bp) = 12V \times 7AH = 84Wh$$

Thus $P_g > B_p$ and is therefore sufficient.

$$Battery\ capacity = Battery\ power \times depth\ of\ discharge = 84 \times 0.7 = 58.8\ Wh$$

C. Temperature and Humidity Sensor (DHT22).

DHT22 sensor was used for measuring temperature and humidity. It is a basic, low cost digital temperature and humidity sensor. It can measure temperature between -40 to 125 degrees Celsius with a 0.5°C accuracy and 0 to 100% humidity at 2 to 5% accuracy. It is easy to use but requires careful timing to achieve the data collection as it only gives new set of data every 2 seconds, which is adequate for this work. The DHT22 sensor has 3 pins of which one is connected to the power supply, the middle pin is the data input and the other pin is for connection to the ground. It has an acceptable

level of precision and accuracy and in a better range of temperature and humidity. It offer excellent quality, fast response, anti-interference ability and cost effectiveness.

The sensor uses a negative temperature coefficient (NTC) thermistor to measure temperature. A thermistor is a variable resistor whose resistance changes as it temperature. For an NTC thermistor, the changes are inversely related, although not linear. To measure humidity, the DHT22 uses a two electrode humidity sensing component with a moisture holding substrate between them. Thus, as the humidity changes, so also does the resistance between these electrodes. The DHT22 sensor is shown in figure 5.



Figure 5. DHT22 Temperature and Humidity Sensor

The DHT22 was interfaced to the Arduino via Digital Pin 4. Algorithm was written to read the sensor values and convert them to appropriate scale using existing libraries and developed sketches. The temperature sensor algorithm to read the temperature from the sensor, convert it to Celsius and display the result on the Arduino IDE serial monitor is shown in figure 6.

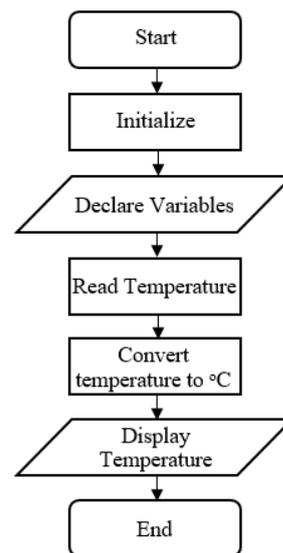


Figure 6. Temperature measurement algorithm

Humidity measurement in the developed system utilized the DHT22 sensor. The algorithm to read the humidity from the sensor and display same on the serial monitor is shown in figure 7.

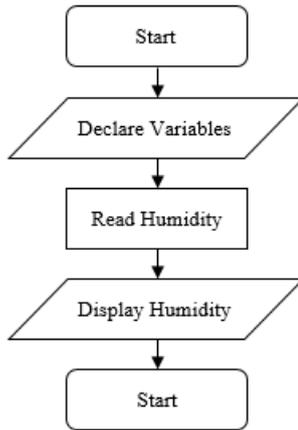


Figure 7. Humidity measurement algorithm

As the same sensor was used for temperature and humidity measurement, the algorithm for humidity sensing did not have the initialization sequence as it was already taken care of.

D. Light Intensity

Light intensity otherwise known as brightness, it is the measure of the wavelength-weighted power emitted by a light source in a particular direction per unit solid angle, based on the luminosity function, a standardized model of the sensitivity of the human eye. S I unit is candela (cd). In this work, the light dependent resistor (LDR) was used to measure the brightness of the sun.

An effective way to measure light intensity is through the use of a lux meter, which is a device that measures the amount of power from the light falling on a given unit of area (lux). Many sensors such as photodiodes, phototransistors, and Light Dependent Resistors (LDR) are used sensors in lux meters. The cheapest, easiest and most readily available for use is the LDR and was therefore used in this work.

An LDR is a device whose resistance changes as the amount of light falling on it changes; the brighter the light, the lower the resistance. Although the relationship between the amounts of light falling on the LDR is inversely proportional to its resistance, the relationship between resistance and lux for an LDR is unfortunately not a direct linear relationship.

A way to solve this non-linear, exponential relationship is to obtain the mathematical relationship between the resistance and the amount of light falling on the LDR and then create an effective lux meter using a microcontroller (Arduino in this work) to program the relationship.

- Characterizing the LDR

We earlier reported in [9] how the LDR was used to achieve a lux meter based on the work by [10]. We reproduce the work here for quick reference. The logarithm equation for the illumination, lux was:

$$\log_{10} lux = -1.4 \times \log_{10} R_{ldr} + 7.098 \quad (1)$$

Where R is the LDR resistance in ohms.

The illumination, lux is obtained from equation 1 as:

$$lux = R_{ldr}^{-1.4} \times 10^{7.098} \quad (2)$$

The LDR was then connected in a voltage divider circuit with a 5 kΩ resistor as shown in figure 8.

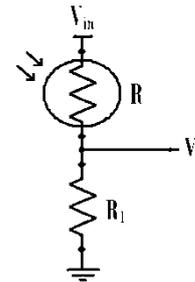


Figure 8. Voltage Divider Circuit for the LDR

The output voltage (i.e. voltage across R₁) is, therefore:

$$V_o = V_{in} \times \frac{R_1}{R_1 + R} \quad (3)$$

Where R is the resistance of the LDR (R_{ldr}), $R_1 = 5 \text{ k}\Omega$ and V_{in} is 5 volts.

The output voltage as read in by the microcontroller is a raw data and hence needs to be converted to its analogue value after finding the LDR Voltage. To accomplish this, the following conversions are used:

The equivalent analogue output voltage, V_a

$$V_a = V_{R1} = \frac{V_o}{1023} \times 5 \quad (4)$$

Therefore, the voltage across the LDR:

$$V_{ldr} = 5 - V_{R1} = 5 - \frac{5V_o}{1023} \quad (5)$$

The LDR resistance is, therefore:

$$R_{ldr} = \frac{V_{ldr}}{V_{R1}} \times R_1 = \frac{1023 - V_o}{V_o} \times 5000 \quad (6)$$

Substituting equation 6, into equation 2 gives the equivalent illumination.

The LDR was interfaced to the Arduino as shown in fig. 9.

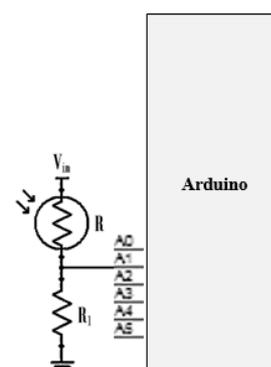


Figure 9. LDR Interfaced as Lux Meter

The lux meter was interfaced via the analogue pin A1 of the Arduino. The output of the voltage divider circuit consisting of the LDR was read as the sensor value by the microcontroller. Conversion and calculations were made on the analogue input to obtain the sun intensity before being displayed on the serial monitor. The flowchart for the measurement of sun intensity is shown in figure 10.

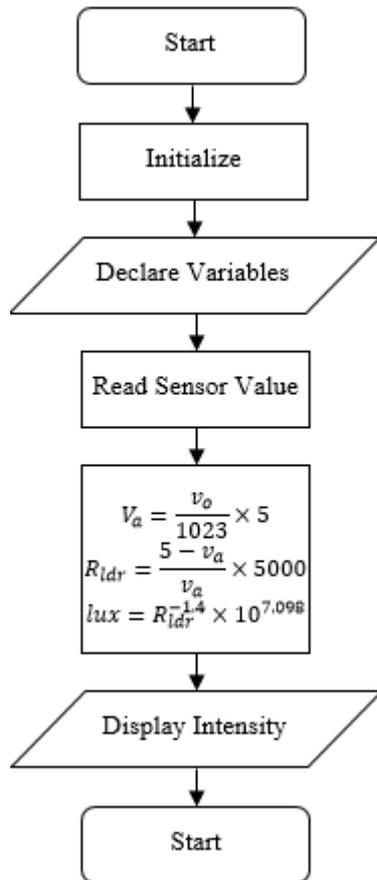


Figure 10. Sun intensity measurement flowchart

E. Wind Speed

Wind speed, or wind velocity, is a fundamental atmospheric rate. Wind speed is the rate of the movement of wind in distance per unit of time. In other word it is the rate of movement of air flow and can be reported as knots or nautical miles per hour, or statute miles per hour. A three-cup anemometer is used to measure wind speed.

An anemometer has a reed switch that is attached to the shaft which is activated at every revolution. The speed of wind can be calculated by measuring the time it takes for a revolution to occur. According to the literature, for a wind speed of 1.492 miles per hour (2.4 kilometers per hour) will result in the reed switch being activated once every second (that is 1 revolution per second). Figure 11 shows the three-cup anemometer [11].



Figure 11. Three – cup anemometer

In order to detect a revolution, a 4.7 kΩ pullup resistor was used to pull the voltage at pin 2 to ground whenever the switch on the cups closes. When the switch was on, the voltage at pin 2 was high, i.e. 5 V. Without the pullup resistor, false revolution trigger could be present as a result of floating voltage of the circuit. With the pullup resistor, when a revolution completed with the close of the switch, the voltage at pin 2 will be pulled to ground for the duration it took the contact to pass the switch. This voltage transition was used to detect a revolution.

The time it took for the revolution to take place was also calculated by subtracting the time for the trigger to occur from the time the last trigger occurred. The present trigger time was stored for future calculation. Using the revolution period, the speed of the wind was determined. Since wind speed varies, the average of returned speed over 2 seconds were used. Figure 12 shows the flowchart for measuring the wind speed.

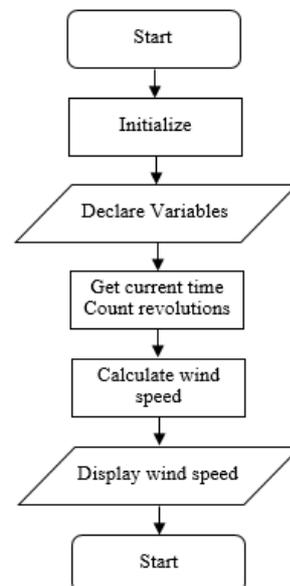


Figure 12. Wind speed measurement flowchart

F. Wind Direction

Wind direction is the direction, measured on the points of a compass, from which wind is coming. The direction of the wind is extremely important in many ways. It is measured by an instrument called a wind vane or weather vane and have been used for indicating which direction the wind is coming from and allow people to attempt to forecast the weather. Figure 13 shows the circuit diagram of the wind vane.

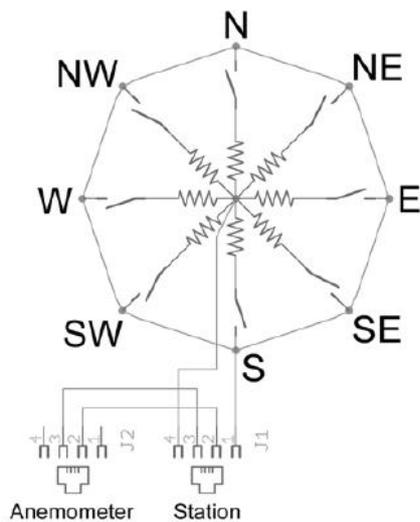


Figure 13. Wind vane circuit

The wind vane has eight switches, each connected to a different resistor. As the vane rotates, it closes a switch. It may also close two switches. The location of the vane is determined using an external resistor to create a voltage divide and then reading the output voltage which is determined by the resistance of the closed switch(es). Since the vane resistances are fixed as well as the external resistance and the supply voltage, the output voltage and hence the direction can easily be determined by the microcontroller (in this case Arduino). Up to 16 different positions can be indicated by the vane. In this work a 5 volt input and 5 kΩ external resistor was used. The Voltage Output V_{out} is shown in table 3. The direction and values of resistors in the vane was obtained from the manufacturers' data in [12].

The switch and resistor arrangement is shown in the diagram to the right. Resistance values for all 16 possible positions are given in the table. Resistance values for positions between those shown in the diagram are the result of two adjacent resistors connected in parallel when the vane's magnet activates two switches simultaneously.

The wind vane was interfaced to the microcontroller as an analogue input on pin A0. Since the 16 possible resistance values were already known as indicated in table 3, the analogue input values were mapped to their corresponding directions. This process was indicated as a flowchart in figure 14.

TABLE III. WIND VANE PARAMETERS

Direction (Degrees)	Resistance (Ω)	V_{out} (V)
0	33000	4.342
22.5	6570	2.839
45	8200	3.106
67.5	891	0.756
90	1000	0.833
112.5	688	0.605
135	2200	1.528
157.5	1410	1.100
180	3900	2.191
202.5	3140	1.929
225	16000	3.810
247.5	14120	3.692
270	120000	4.800
292.5	42120	4.469
315	64900	4.642
337.5	21880	4.070

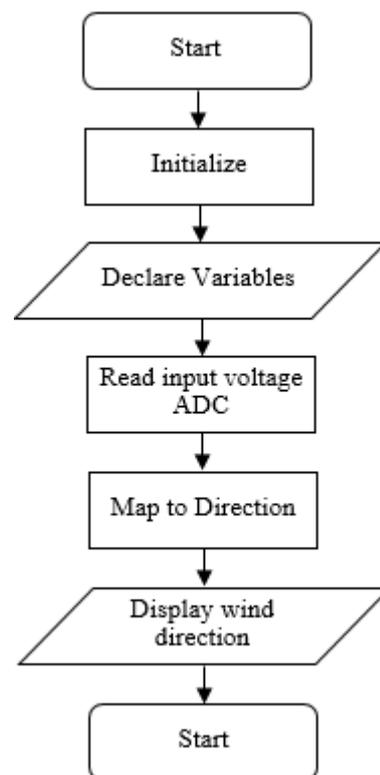


Figure 14. Wind direction detection flowchart

G. Rain Gauge

This is a tipping bucket type that self-empties itself. It consists of a funnel that channels the water into one of two buckets that are made to as a seesaw. When 0.2794 ml of water is collected in one of the buckets, it tips and pour away the

water, while the other bucket is placed to collect the water [12]. As the bucket tips, a magnetic switch acts to send an electrical signal to indicate tipping. Thus the cumulative rain water collected over a period of time can be measured. The rain gauge used is shown in figure 15.



Figure 15. Rain Gauge

The rain gauge was interfaced via the digital PIN 2 of the Arduino. Whenever the magnetic switch action was recorded, the quantity of rain water collected was incremented by 0.2794 ml. The tipping time was recorded against the current volume of water collected. At midnight, the volume of collected water for the day was recorded and reset to allow for the next days' measurement. The rainfall measurement algorithm is shown in figure 16.

The code was such that an hourly reading was returned for a day (24 hours) before the reset. Since all readings were sent to the data logger, a reset would not affect the rainfall amount reading over a long period.

H. XBee Module

The XBee (Series 1) RF Module is a low-cost and low-power wireless module that meets the IEEE 802.15.4 standards. It delivers data reliably between devices while operating within the ISM 2.4 GHz frequency band and are pin-for-pin compatible with each other, having a line-of-sight range of up to 100 meters and requires a voltage supply of 3.3 volts [13].

A series 1 XBee is used for the purpose of this work as it costs less and is easier to use. It comes preconfigured; so it can be used without further configuration provided no other similar device operates within its range. The XBee module and its pin layout are shown in figure 17.

In order to reconfigure the XBee's default settings and also handle the desired supply voltage, an XBee explorer is required. The XBee explorer is a reliable USB interface for XBee modules, for configuration or communication with PC, shown in figure 18. This unit works with the various series of the XBee modules plugging the unit into the XBee explorer, attaching USB cable to get a direct access to the serial and programming pins on the XBee unit.

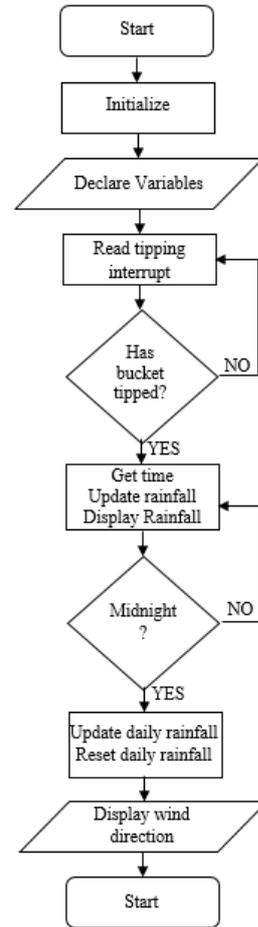


Figure 16. Rainfall measurement algorithm

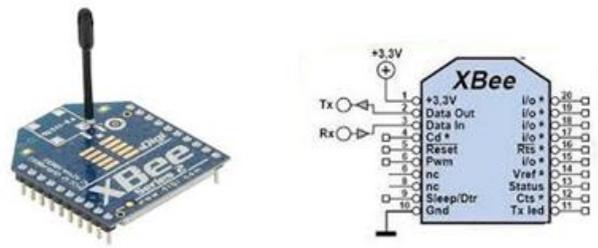


Figure 17. XBee Module



Figure 18. XBee Explorer

Using the XBee Explorer USB, the XBee was connected to the PC after installing all necessary drivers. The XCTU Software for windows was downloaded and installed in order to reconfigure the XBee. After selecting the correct COM port for the XBee available on the PC Settings of the XCTU application. Information about the XBee was accessed by clicking on the Test/Query button. The Modem Configuration tab was then opened and the current XBee configuration was accessed by clicking on the Read button. The XBee channel and network id were changed from the default values. The procedure was repeated for the second XBee, ensuring that both operate over the same configurations except for the CE, in which the receiver was set to “coordinator” while the transmitter to “end point”.

The received weather parameters were available on the serial port of the Arduino with the receiver XBee. All that was needed was an application that will open the serial port and read the received parameters. MySQL connector for Arduino was downloaded from [14] and used for the purpose.

The designed system is complete Automatic weather station for remote monitoring. The transmitting microcontroller was programmed to send the parameters to the receiving microcontroller via the XBee modules and saved to the computer system at the rate of one string every 30minutes, and also to display the same parameters on the LCD display. The computer system receives the string of data and was processed by a software application using MySQL. This database displays the current parameters on the screen of the computer system which has data logger functions. The received data was sorted according to each parameter being measured and displayed accordingly on a walking graph. With each new value received, the graph is updated to correspond to value and time. The values were also saved and added to the existing each time a new transmission was received.

1) Vero board implementation

The components were soldered on a VERO board. Figure 19 shows the cased implemented system.



Figure 19. Vero Board Implementation

2) Testing

An existing weather station was made available for use as the ground truth (WSg). The components of the WSg was compared to the developed weather station (WSd) as shown in table 4.

TABLE IV. COMPONENTS OF THE GROUND TRUTH AND DEVELOPED WEATHER

Measured Parameter	Weather Station	
	Ground Truth WSg	Developed WSd
Solar radiation	Campbell stoke recorder	Lux meter
Temperature measurement	Thermometer	DHT22
Rain measurement	Rain gauge	Rain Gauge
Humidity	Hygrometer	DHT22
Wind direction	Wind vane	Wind Vane
Speed of wind	Anemometer	Anemometer
Data transmission	GSM module	ZigBee
Saving measured parameters	-	Datalogger

The developed weather station was placed about 1 meter beside the ground truth weather station. The stations were about 10 meters away from the location of the data loggers. Measurements were obtained for both stations over 4 hours.

Temperature measurement in the ground truth was through the use of a standard thermometer. The measured temperature was therefore subject to human errors such as parallax which are unavoidable in reading manual thermometers. The temperature measured by the DHT22 by the developed system was compared with the thermometer readings from the ground truth for each period of recording. The humidity values obtained by the developed system and the ground truth station using a hygrometer were compared.

The value of the intensity of sunlight was recorded at the datalogger after being sent through the XBee from the developed weather station. The intensity obtained from the ground truth was then compared with that of the developed system for each time recorded. The ground truth measurement was performed using a Campbell stoke recorder while the developed system was made using the developed lux meter.

An anemometer was used at both stations to measure the speed of wind. It rotates at the same speed as the wind which gives direct measure of the speed of the wind. Beaufort wind scale is used to measure wind speed which is a scale of 0-12 based on visual clues. A three-cup anemometer was available and was used in this work to measure wind speed.

As the test period was not a rainy season, to evaluate the developed rain gauge, a clinical 5ml syringe was used. This gave us the ability to predetermine the quantity of water to expect the rain gauge to record.

IV. RESULT AND DISCUSSION

The results obtained from individual units and collective system is discussed in this section.

A. Rain

Test period was not a raining season, thus, two 5 ml syringes were used to replicate rainfall and to be able to know the exact water volume that was discharged unto the rain gauge. Knowing the water volume enabled us to know the water volume to expect from the weather station. The syringe therefore served as a control. The timing was also not the same as the times for other variables due to the manual approach in simulating the rainfall with the syringe. The obtained simulated rainfall using the syringe is given in figure 20.

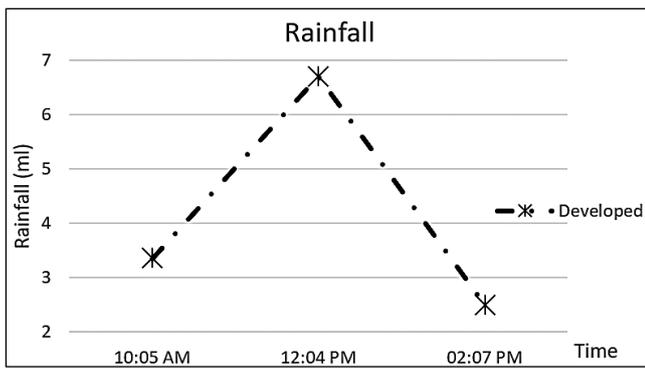


Figure 20. Rainfall

Because the tipping bucket in the rain gauge only tips after 0.2794 ml of water was collected. At 10:05 am, a total of 3.5 ml of water was discharged from a 5ml syringe. The rain gauge was only able to register 3.3528 ml giving an error of 0.1472 ml. This was within the error margin of the tipping bucket which required 0.2794 ml of water to be collected before tipping over and making a count toward the total collected water. Since in this case the error was 0.1472 ml, which was less than the 0.2794 ml of water required for tipping. This translates to an error of about 4.2 %. It can therefore be concluded that the remaining water was in the bucket but was not part of the count as it was not enough to tip and be included in the volume collected. We can also conclude that the tipping bucket will have an error of up to 0.27 ml. To counter this error, an average over a longer period may be used. This will also come with its shortcoming especially in areas where rainfall can be erratic; high within a period and suddenly absent or very low the next.

B. Wind Direction

The wind direction indicated were the same in both the developed weather station as well as the one on the ground truth. The readings are shown in table 5.

TABLE V. WIND DIRECTION (DEVELOPED VS. GROUND TRUTH)

Time	03/06/2016		
	10: 00 am	12:00 pm	2:00 pm
Developed	NEE	NEN	SE
Ground Truth	NEE	NEN	SE

There were just 16 points and therefore approximations were made such that a given direction had some angular range from the anemometer analog input. This range could have been responsible for cancelling out possible difference between the two readings.

C. Wind Speed

The wind speed varied between the one obtained from the developed station as against that from the ground truth as shown in figure 21.

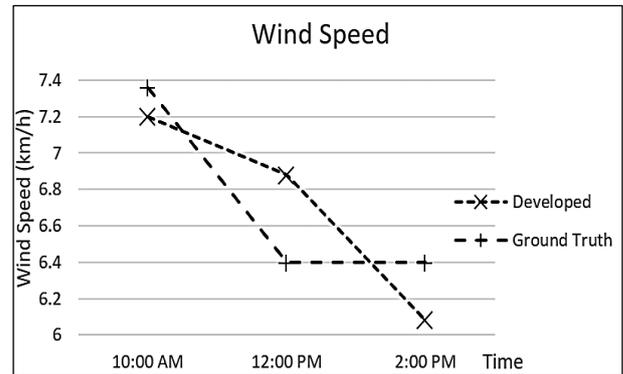


Figure 21. Wind Speed (Developed vs. Ground Truth)

The variation doesn't seem to emanate from one system. As can be seen from the table, at 10:00 am, the wind speed was 7.20 and 7.36 km/h respectively in the developed and the ground truth resulting in the ground truth recording a 0.16 km/h faster wind speed than the developed system. At 12:00 pm, the developed system and the ground truth recorded wind speeds of 6.88 and 6.40 km/h respectively. This translated a 0.48 km/h faster wind recorded by the developed system. At 2:00 pm, the wind speed was 6.08 and 6.4 km/h in the developed and the ground truth respectively, resulting in the ground truth recording a 0.32 km/h faster wind speed than the developed system. A correlation of 72.05% was obtained between the wind speed values recorded by the developed system and the ground truth.

A possible issue resulting in the mismatch between the two could be the averaging of wind speed over 2 seconds in the developed system or the approach in the ground truth which is not known to us yet.

D. Humidity

The humidity obtained from the developed system and the ground truth is shown in figure 22.

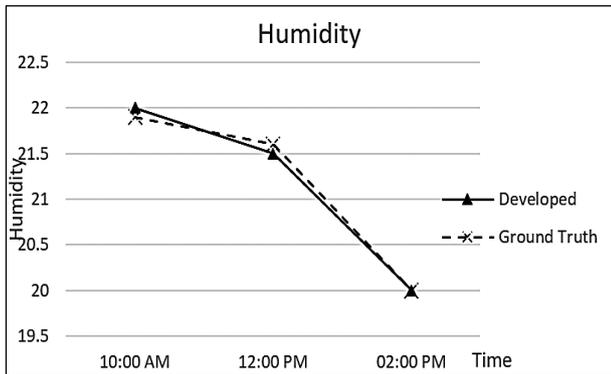


Figure 22. Humidity (Developed vs. Ground Truth)

The humidity value was 0.1 higher and 0.1 lower at 10:00 and 12:00 pm respectively in the developed system as against the ground truth. The two systems had the same values at 2:00 pm.

A 99.54% correlation between the humidity values measured by the developed and the ground truth was obtained.

E. Temperature

The temperature readings for the developed system had 2 digit accuracy as against that of the ground truth that was always approximated to the nearest whole number. This was because as already stated; the ground truth had a manual thermometer which made it impossible to have precise readings due to parallax errors. The temperature obtained from the Developed system and the Ground Truth is given in figure 23 and figure 24.

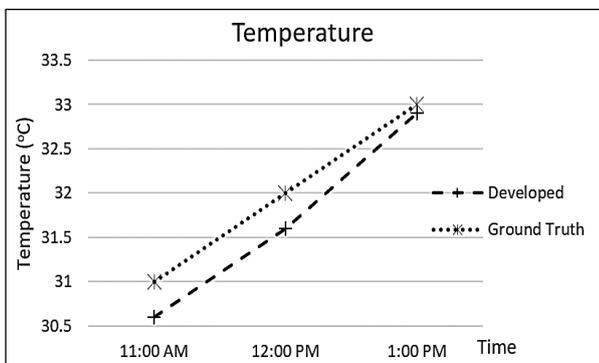


Figure 23. Temperature (Developed vs. Ground Truth)

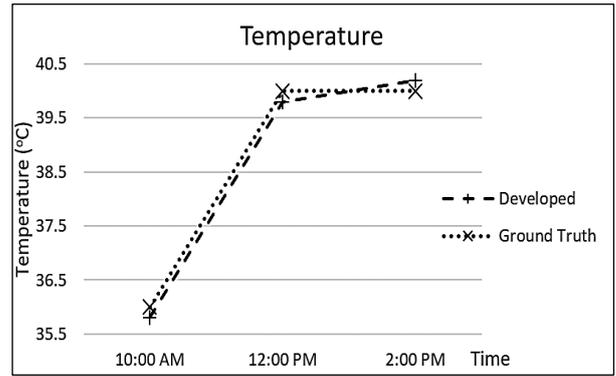


Figure 24. Temperature (Developed vs. Ground Truth)

For the first evaluation, a correlation of 99.66% was obtained between the temperature values recorded by the developed system and the ground truth. During the second evaluation, a correlation of 99.72% was obtained between the temperature recorded by the developed system and the ground truth.

F. Voltage Level

The voltage level was returned at periodic intervals alongside the weather values. A sample of returned voltage level is given in figure 25.

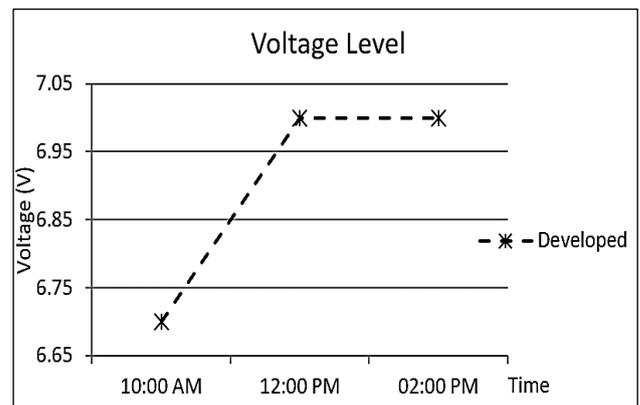


Figure 25. Voltage Level (Developed)

The voltage level increased from 6.6 volts at 10:00 to 7.00 volts at 12:00 pm and stayed the same at 2:00 pm. At 10:00 am, the system was being charged by the solar unit. At 12:00 pm, it was fully charged and stayed the same by 2:00 pm. During no-sun period such as at night or cloudy periods, the battery will start to discharge as the system draws power from it; since the battery at that period does not have power to replenish the drawn power.

V. CONCLUSION

A remote weather monitoring system to monitor temperature, humidity, wind speed, wind direction, sunlight intensity and rainfall was developed. The system was powered using a solar panel and battery for standalone operation and was able to display the battery voltage. The temperature, humidity, wind speed and wind direction modules were evaluated against an existing system and correlation of 75%, 99% and 99% were obtained for the wind speed, humidity and temperature variables between the developed and existing systems. In the wind direction monitoring, a 100% resemblance between the developed and existing system was obtained. Rainfall was simulated using a syringe of known volume and the system was able to record it correctly with 96.8% accuracy. The measured data was transmitted and received by XBee modules and thereafter saved to a PC.

REFERENCES

- [1] K. Krishnamurthi, S. Thapa, L. Kothari and A. Prakash, "Arduino Based Weather Monitoring System," *International Journal of Engineering Research and General Science*, vol. 3, no. 2, pp. 452 - 458, 2015.
- [2] K. Nandagiri and J. R. Mettu, "Implementation of Weather Monitoring System," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 16, pp. 477 - 493, 2018.
- [3] B. Mihai, "About the Smart Weather Station," *ACTA Universitatis Cibiniensis*, vol. 68, no. 1, pp. 26 - 29, 2016.
- [4] D. R. Brooks, "Links to Arduino-Related Documents," 30 03 2018. [Online]. Available: <http://www.instesre.org/ArduinoDocuments.htm>. [Accessed 21 03 2019].
- [5] P. J. Sousa, R. Tavares, P. Abreu and M. T. Restivo, "NSensor – Wireless Sensor Network for Environmental," *International Journal of Interactive Mobile Technologies*, vol. 11, no. 5, pp. 25 - 36, 2017.
- [6] A. Katyal, R. Yadav and M. Pandey, "Wireless Arduino Based Weather Station," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 5, no. 4, pp. 274 - 276, 2016.
- [7] A. U. Adoghe, S. I. Popoola, O. M. Chukwuedo, A. E. Airoboman and A. A. Atayero, "Smart Weather Station for Rural Agriculture using Meteorological Sensors and Solar Energy," in *Proceedings of the World Congress on Engineering 2017*, London, United Kingdom, 2017.
- [8] P. Susmitha and G. Sowmyabala, "Design and Implementation of Weather Monitoring and Controlling System," *International Journal of Computer Applications*, vol. 97, no. 3, pp. 19 - 22, 2014.
- [9] A. H. Alkali, E. G. Dada, A. M. Kida and A. A. Ali, "Sunlight and Rainfall Activated Retractable Roof," *International Journal of Computer Engineering and Applications*, vol. 12, no. 11, pp. 1 - 12, 2018.
- [10] D. Williams, "All About Circuits," 13 12 2015. [Online]. Available: <https://www.allaboutcircuits.com/projects/design-a-luxmeter-using-a-light-dependent-resistor/>. [Accessed 06 08 2018].
- [11] N. Sabharwal, "A Low Cost Zigbee Based Automatic Wireless Weather Station with GUI and Web Hosting Facility," 2014.
- [12] Argent Data Systems, "Wind/Rain Sensor Assembly," [Online]. Available: www.argentdata.com/files/80422_datasheet.pdf. [Accessed 02 03 2019].
- [13] Digi International Inc., "Product Manual v1.xEx - 802.15.4 Protocol," Digi International Inc., Minnetonka, 2009.
- [14] C. Bell, "MySQL Connector/Arduino," [Online]. Available: [www.github.com/ChuckBell/MySQL_Connector_Arduino](https://github.com/ChuckBell/MySQL_Connector_Arduino). [Accessed 13 01 2019].

How to Cite this Article:

Alkali, A. H., Bassi, S. J., Dada, E. G., Bukar, Z. M., Idris, M. & Abdulfattah, M. H. (2019) Design, Implementation and Evaluation of Remote Weather Monitoring System. *International Journal of Science and Engineering Investigations (IJSEI)*, 8(87), 65-76. <http://www.ijsei.com/papers/ijsei-88719-10.pdf>

