A Generic Configurable Plug-and-Play Module for Node Data Acquisition and Transmission in IoT-based Applications

Nkolika O. Nwazor [1], Rosaline Ashinedu Eke [2], Ekene Samuel Mbonu [3]

[1] Electrical/Electronic Engineering Department University of Port Harcourt, River State, Nigeria
[2] Centre for Information and Telecommunication Engineering, University of Port Harcourt, River State, Nigeria
[3] Mechatronics Department, Federal University of Technology Owerri, Imo State, Nigeria

Abstract

In this era of advancements in IoT technology, the development of a commercial, configurable module for easy node data acquisition and transmission in IoT applications is a necessity. Most of the available systems are application-specific. This research work is on the development of a plug-and-play module for node data acquisition and transmission in IoT-based applications. A generic framework for IoT sensors and controllers was developed. In this work, a plug-and-play device that adapts non-IoT sensors and controllers for use in IoT applications was developed. Software application for the implementation of the model was developed. Online validation of the work using an open-source server was carried out. The node can be used to acquire data from any remote location that has a GPRS network. Therefore, the system can be applied in many areas of IoT automation for remote data acquisition. The throughput of the system is 95%. The latency of the node was found to be 3 seconds, while the processing time of the node is 135 seconds. The system proposed in this research will reduce the design to market time for IoT systems to two hours. This work will help in the standardization of the backend of remote IoT nodes for easy integration in the larger IoT ecosystem. The work can be commercially produced as generic configurable IoT controller modules by original equipment manufacturers (OEMs) thereby making it simpler to apply IoT technology in several fields of endeavours.

Keywords: Internet of Things, Plug-and-play, Data Acquisition, Data Transmission, IoT controllers, Sensors, Networks

Introduction

In recent times, there has been a great increase in the number of devices connected to the internet. About 35 billion Internet of Things (IoT) devices are connected to the internet and there is a projection that this number will increase to about 120 billion by the year 2025 generating about 180 billion terabytes of sensors [1]. Internet of Things (IoT) which is a major technology supporting industry 4.0 has some challenges when it comes to remote applications. A 'thing' in the IoT domain is supposed to perform at least two functions: data acquisition and transmission. Data acquisition, which usually entails reading and calibrating physical parameters, is usually application-specific making it difficult to have a universal IoT node for sensor data acquisition and transmission within

the IoT ecosystem [2]. Thus, IoT nodes before now are usually application-specific. The implication is that anybody that wants to use an IoT node must have to design and fabricate the required specific IoT node for the task at hand. For example, in a system that requires temperature, humidity, pressure, air quality monitoring at different locations, separate IoT nodes with unique algorithms have to be designed and fabricated for each parameter to be monitored.

This does not make for easy and quick implementation of IoT systems. This work aims to develop a universal hardware and software framework for IoT node data acquisition and transmission. This framework will accept sensor readings as inputs while using the universal software framework to achieve automatic sensor calibration, and data transmission using the supplied receiver's address. Basic IoT systems monitor and control physical variables that define the "things" they monitor and control. To design an IoT system, the physical quantities that require monitoring must be defined [3]. The design of the data acquisition system is based on this definition. The acquired data must be shared over the Internet with the following constraints:

A) minimization of the amount of the exchanged data while maintaining the entirety of the information
B) reduction of the time delay required for sharing the data over the network, and an increase in the lifetime of the nodes

Some authors have worked on data acquisition and transmission in IoT. The authors in [4] developed a custom IoT data acquisition system. In their approach, raspberry pi was used to demonstrate how a specific sensor can be interfaced to data acquisition hardware. The work gave an insight into how to develop an IoT sensor node. However, it is not a universal product that can be pushed into the IoT market as a solution for data acquisition and transmission at the backend. The work done by [5] is much similar to that of [4]. The major difference is that it was specifically for temperature monitoring. The work by [2] confirmed that before now IoT nodes have always been application-specific. In their approach, an indoor air quality monitoring platform was developed using some selected air quality sensors.

A non-invasive, data acquisition system for reliable bio-telemonitoring of pregnant women which used National Instruments (NI) myRIO for data acquisition and transmission of the acquired physiological signals wirelessly to a computer running NI LabView software for real-time signal visualization, processing and data logging was presented in [6]. The authors in [7] used NodeMCU to push medical parameters to the ThingSpeak, and also Message Queuing Telemetry Transport (MQTT) was used to push data to the server, and the REpresentational State Transfer (REST), while web services were used to provide interoperability between computer systems on the Internet.

The authors [8] designed a re-configurable smart sensor interface for industrial wireless sensor networks in an IoT environment that made use of the complex programmable logic device (CPLD) as the core controller. Scientech also developed their 6205DA IoT Data Acquisition System and Protocol Converters which is a platform for exploring architecture, working and design applications of a data acquisition system [9]. The system understands types of protocol converters like serial to Ethernet converter, serial to Wi-Fi converter, and serial to GPRS. In [10] a generic Internet of things

(IoT) platform supporting plug-and-play device management was developed based on the semantic web. This work is a major shift from the application specific devices proposed by majority of the authors that have worked in this area. One of the authors used GSM as the only gateway for data transmission. There is no generic plug-and-play device for adapting non-IoT compliant sensors and controllers to IoT environments that has the capability of using multiple data transmission gateways.

Materials and Methods

The block diagram of the proposed model for data acquisition and transmission in IoT-based applications is shown in figure 1. This work proposes that just three plug-and-play components are needed to implement a typical IoT remote node. The components are IoT sensor, IoT controller with configurable software, and IoT gateway. The goal is to develop a system that when implemented will produce three off-the-counter IoT components that can be configured to acquire and transfer data to any addressed receiver via the IoT gateway.

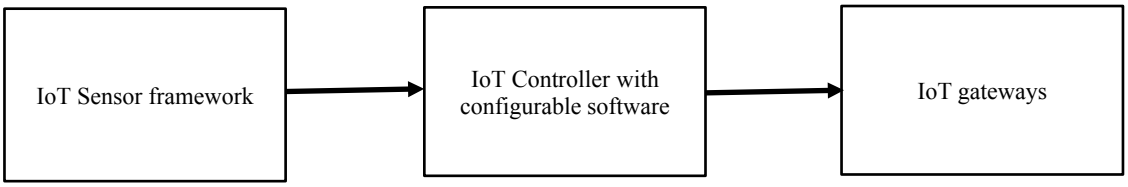
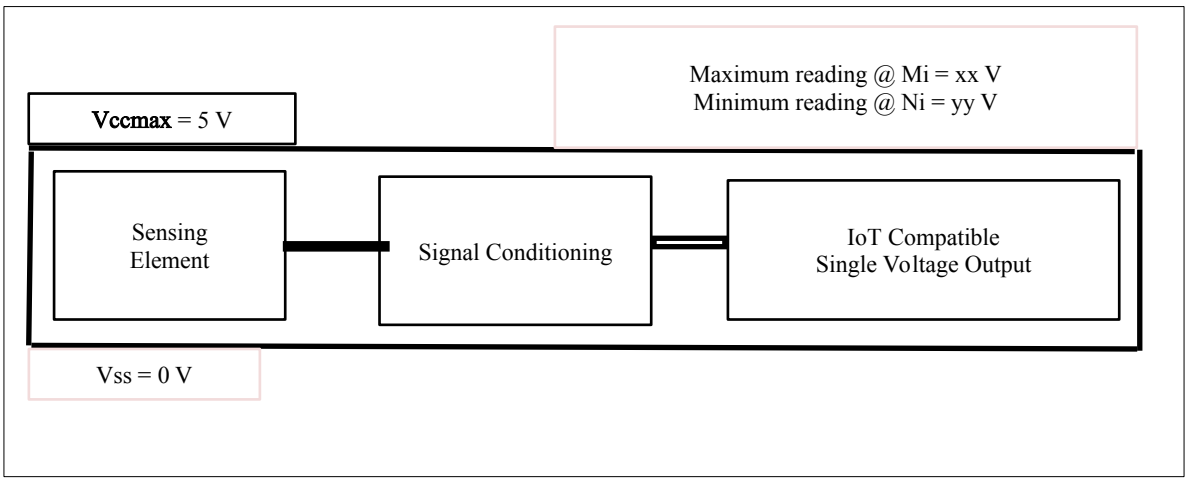


Figure 1: Block Diagram of the Proposed Model

A. Description of the Sensor Framework

The description of the proposed IoT sensor framework is shown in figure 2.

Figure 2: Abstraction of IoT sensor framework



By this abstraction of figure 2, it is being proposed that for any sensor to be classified as an IoT sensor, the following condition must be satisfied:

- Its power supply must have a range between 0 and 5 V.

- It must have a signal conditioner that normalizes its output signal to 0-5 V.
- Maximum reading xx of the sensor at maximum physical parameter M_i must be stated. For example, if the highest temperature the sensor can measure is 100° C, then the equivalent voltage reading of the sensor at 100° C must be stated by the original equipment manufacturer (OEM)
- Minimum reading yy of the sensor at minimum physical parameter N_i must be stated.

With this narrative, it becomes clear that non-IoT-based sensors like thermistors, light-dependent resistors (LDR), and other several sensing elements cannot be classified as IoT sensors since they did not meet up with the above abstraction/standard. However, an original equipment manufacturer, OEM can use an LDR as a sensing element, for example, to produce a standard IoT light sensor. The same applies to thermistors. It can be used to produce a standard IoT temperature sensor. The whole idea of the sensor framework is to have a standard for IoT sensors such that any of such sensors can be interfaced directly to any IoT controller.

B. Description of the IoT Controller Framework

To have an easy interface with IoT sensors and IoT communication gateways, an abstraction of a universal IoT controller is proposed in figure 3. The implication of the IoT controller framework abstracted in figure 4 is that for a controller device to be classified as an IoT controller, it has to satisfy the following conditions:

- It must have N channel sensor input ports that can accept analog inputs from any IoT sensor.
- It should have an N-1 multiplexer that will help the data acquisition software classify the read sensor data.
- It should have an inbuilt analog to digital converter.
- It should have inbuilt configurable data acquisition and communication software.
- It should have a provision for a computer interface so that IoT experts can configure it.
- It should have gateway interfaces that will enable the controller interface with Wi-Fi, Bluetooth, GPRS as well as SMS gateways as occasion demand

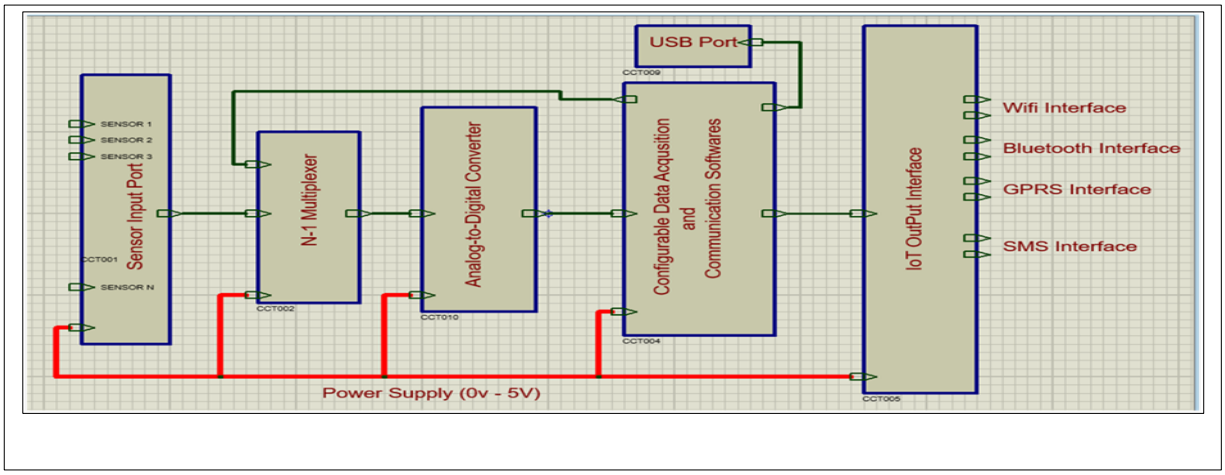


Figure 3: The Abstraction of an IoT Controller Framework

Again, with the above abstraction, it is clear that many microcontrollers like AT89C51, ATmega328, etc., cannot be classified as IoT controllers. Instead, OEM can pick up these microcontrollers, and then use them to design IoT controllers. Once IoT sensors and IoT controller becomes a commercial commodity, then one can buy and use them in IoT application. It will just be a question of configuring the controller appropriately.

C. Description of IoT Gateways

IoT gateways are special communication modules designed to interface with microcontrollers. Most of them are already transistor-transistor logic compliant, so this work will not dwell much on the gateways.

D. Model for Data Acquisition and Transmission in the Proposed Framework

Consider the string of data, Str_d to be transmitted through the controller gateway of the IoT node. The data can be abstracted to be a function of sensor inputs $S_{1\text{-}n}$, and the gateway transmission protocol G_i. So mathematically,

$$Str_d = F\left(S_{1-n}, G_{i,}\right) \tag{1}$$

It means the string of data is a function of two variables.

(1) The Mathematical Representation of the Sensors Input to ADC

Let the sensors input to the controller with an n-channel analog-to-digital converter (ADC) as shown in figure 4 be S_{1-n}. If the analog equivalent of the sensors' input after going through the ADC is given by S_a, then S_a can be represented as shown in equation 2:

$$S_a = F\left(S_1, S_2, S_{3,}, \ldots, S_n\right) \tag{2}$$

Where f is the ADC's conversion transfer function
For n-bit ADC,

$$f = \frac{V}{2^n} \tag{3}$$

Where V is the maximum dc voltage applied to the ADC.
For a single sensor input, we can write that

$$S_a = S_{a1} \tag{4}$$

Where $S_{a1} = F\left(S_1\right)$ and S_1 is the IoT sensor's reading for sensor 1.
Let $\oplus$ be the factor that concatenates the sensor variables. It follows then that

$$S_a = S_{a1} \oplus S_{a2} \oplus S_{a3} \oplus \ldots \oplus S_{an} \tag{5}$$

(2) Automatic Calibration of the Sensors Output

Equation 5 is the concatenated analog equivalent of the ADC's output. It is not calibrated. Let the reading of sensor 1 at the maximum physical parameter be x_1, while the minimum y_1 be the minimum reading of sensor 1. Automatic calibration of the sensor can be achieved by using a mapping function such that

$$S_{a1c} = \text{map}\ (S_{a1}, 0, 2^n, y_1, x_1) \tag{6}$$

S_{a1c} is the calibrated output of the sensor 1 reading S_{a1}

Finally, if the sensor data to be transmitted through the IoT gateway is given by S_{aT} it follows then that

$$S_{aT} = S_{a1c} \oplus S_{a2c} \oplus S_{a3c} \oplus \ldots \oplus S_{anc} \tag{7}$$

For equation 6 to be implemented successfully, the OEM must supply y_1 and x_1

(3) Selection of Transmission Algorithm

Four gateways are proposed in this work for IoT data transmission: Wi-Fi, Bluetooth, GPRS, and SMS gateways. All of them must not necessarily be used at the same time. The user can decide which gateway to be used by configuring the IoT controller appropriately using equation 8.

G_i = WiFi for i = = 1
= bluetooth for i = = 2
= GPRS for i = = 3
= SMS for i = = 4 (8)

G_i can further be expressed as

$$G_i = G_1 + G_2 + G_3 + G_4 \tag{9}$$

$i = 1, 2, 3, 4$
Where
G_1 = Wifi transmission algorithm
G_2 = Bluethoot Transmission algorithm
G_3 = GPRS Transmission algorithm
G_4 = SMS Transmission algorithm

Meanwhile, the plus sign in equation 9 can be treated as OR logic, so depending on the value of i that is selected during the controller configuration, the appropriate gateway can be selected.

The implication of equation 9 is that every IoT controller should have inbuilt G_i transmissions algorithms.

Hardware System Implementation

A. IoT Sensor Framework Implementation

To test the proposed model, the IoT sensor framework was implemented using a case study of light and temperature sensors. The IoT light sensor was designed using a light-dependent resistor (LDR) as a sensing element. The IoT controller architecture was implemented using ATmega328 as the core controller. The pinouts of the IoT architecture are shown in fig.4. It has six sensor channels and four gateway channels: Bluetooth, WIFI, GPRS, and SMS channels. The power supply to the controller is an independent unit.

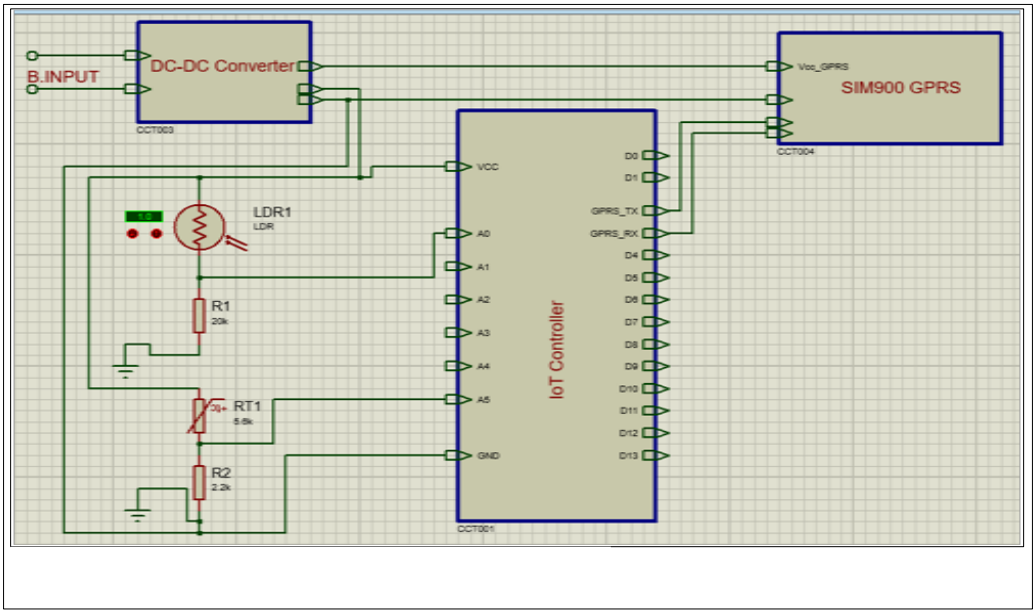


Fig.4: Circuit Diagram of the Integrated Hardware

B. The software Design

The agile method of software development was used to develop the software embedded in the controller. First, the sensor channel configuration software (SCCS) was developed and tested. Next data computation and automatic calibration software (DCACS) was developed using SCCS output as its input. Finally, the data transmission software (DTS) was developed, and also tested. Equations 5 to 9 were used to develop the software. Figure 5 is the high-level flowchart of the software application

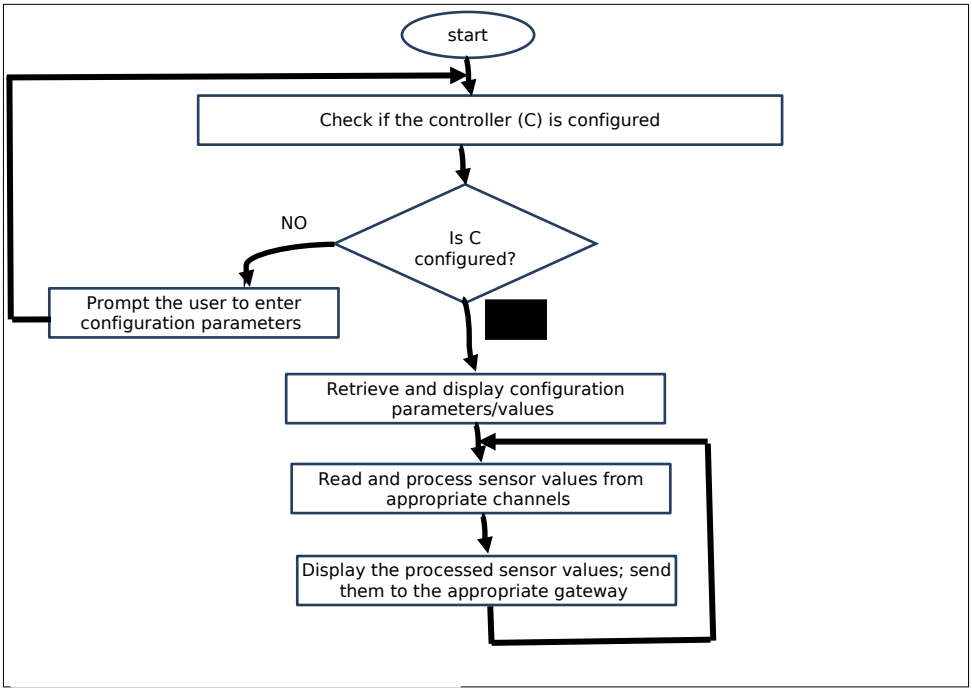


Figure 5: Flowchart of the Software Application

C. Data Collection Plan

The generalized form of equation 6 is given by equation 10:

$$S_{aic} = \text{map}(S_{ai}, 0, 2^n, y_i, x_i) \quad (10)$$

To validate the concept presented in this work, one of the IoT sensors designed and fabricated was used to collect data for $i=0,1,2,3,4,5$. This is to prove that when the concept of this work is commercialized, each sensor channel of the IoT controller can be configured to match any available IoT sensor as proposed in this work.

Results

Table 1 is the output of the UIoTSn when the two IoT sensors designed in this work were connected to channels 0 and 5. The light IoT sensor was connected to channel 0 while the IoT temperature sensor was connected to channel 5. The light intensity on the light sensor was varied by covering the sensing element with hand to a varying degree. Table 1 is the GPRS output of the UIoTSn which was sent to the online ThingSpeak server while Figure 7 and 8 are the snapshots of the online server outputs

Table 1: GPRS output of the UIoTSn

S/N	i = 5; s0, s1, s2, s3, s4, s5
1	1000.00,0.00,0.00,0.00,0.00,32.00
2	1000.00,0.00,0.00,0.00,0.00,32.00
3	1000.00,0.00,0.00,0.00,0.00,32.00
4	1000.00,0.00,0.00,0.00,0.00,32.00

5	1000.00,0.00,0.00,0.00,0.00,32.00
6	1000.00,0.00,0.00,0.00,0.00,32.00
7	1000.00,0.00,0.00,0.00,0.00,32.00
8	1000.00,0.00,0.00,0.00,0.00,32.00
9	1000.00,0.00,0.00,0.00,0.00,32.00
10	1000.00,0.00,0.00,0.00,0.00,32.00
11	1000.00,0.00,0.00,0.00,0.00,32.00
12	1000.00,0.00,0.00,0.00,0.00,32.00
13	1000.00,0.00,0.00,0.00,0.00,32.00
14	1000.00,0.00,0.00,0.00,0.00,32.00
15	1000.00,0.00,0.00,0.00,0.00,32.00
16	124.00,0.00,0.00,0.00,0.00,33.00
17	No data received
18	1000.00,0.00,0.00,0.00,0.00,32.00
19	1000.00,0.00,0.00,0.00,0.00,32.00
20	77.00,0.00,0.00,0.00,0.00,33.00

Table 2: Output Data from the Universal IoT Sensor Node (UIoTSn)

S/N	i = 0; s0	i = 1; s0, s1	i = 2; s0, s1, s2	i = 3; s0, s1, s2, s3	i = 4; s0, s1, s2, s3, s4 =	i = 5; s0, s1, s2, s3, s4, s5
1	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
2	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
3	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
4	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
5	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
6	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
7	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
8	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
9	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
10	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
11	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
12	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
13	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
14	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
15	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
16	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
17	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
18	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
19	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00
20	32.00	0.00,32.00	0.00,0.00,32.00	0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,32.00	0.00,0.00,0.00,0.00,0.00,32.00

A. Universal IoT Sensor Node Functionality

Any of the input channels of the UIoTSn, which serves as a Proof of Concept (POC), can be configured to accept IoT sensors' output. The results showed that the temperature readings of the IoT sensor were consistent with all the channels. Also from Table 1 and other generated data, it can be seen that the UIoTSn can be configured to accept IoT sensors with different bandwidths of maximum and minimum readings without interfering with one another's readings. The UIoTSn was

able to send data to the remote online server. What it means is that this node can be used to acquire data from any remote location that has a GPRS network, and a good enough GPRS network covers a wide geographical range just as a GSM network. The implication is that the UIoTSn can be applied in many areas of IoT automation for remote data acquisition.

(1) Throughput of the UIoTSn

Throughput in this work can be defined as the ratio of data sent from the UIoTSn to that of the data successfully received at the server end. A total of 20 sampled data were sent. There was only one failure. So, the throughput of the UIoTSn in percentage is

$$\left(\frac{19}{20}\right)\times 100 = 95\%$$

This is much acceptable for any data acquisition system!

(2) Latency of the UIoTSn

The latency of the system is defined as the time it takes data to move from UIoTSn to the online server. Experimentally the latency of the node was found to be 3 seconds, while the processing time of the node was 135 seconds.

(3) The integrity of the UIoTSn

The integrity of the data acquisition system has to do with delivering the exact data acquired by the system. A comparison of data sent with that received showed that the UIoTSn has 100% integrity.

(4) System Validation

One of the major achievements in this work is the automatic calibration of sensor readings which usually takes time for any data acquisition system. But this work overcame the challenge by specifying the maximum and minimum readings of the IoT sensor nodes, a method being proposed that IoT sensor node OEMs should adopt. So the maximum and minimum readings of the developed IoT temperature sensor were supplied as part of the inputs to the configuration software. The temperature output was compared with that of a standard analog thermometer placed within the same environment as the IoT temperature sensor. While the thermometer read 32.2° C, the UIoTSn measured 32° C, giving a difference of 0.2 i.e 99.4% accuracy. As proof of concept for the data acquisition and transmission system, the designed integrated system was implemented.

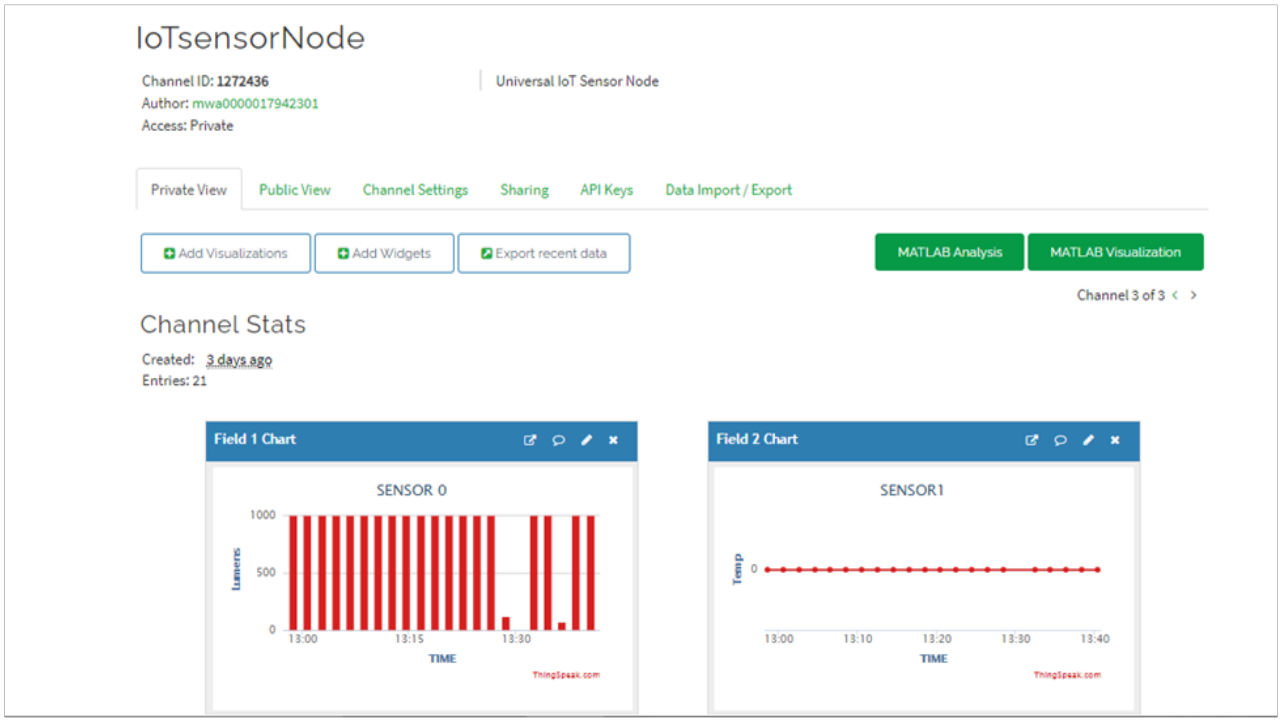


Figure 6: Snapshot of the Online Server Output of the UIoTSn

B. Software Implementation

The SCCS and DCACS were integrated and tested. The data were sampled at intervals of 2 seconds. The data transmission software (DTS) was developed for the case of $G_i=3$ due to the time factor. To test the DTS, a real-time online server was configured using the ThingSpeak open-source server.

Conclusion

The objectives of this research work were achieved successfully. The model for IoT data acquisition and transmission was developed. Software applications were developed and the results from the test were within acceptable limits. The system can be configured to accept IoT sensors with different bandwidths of maximum and minimum readings without interfering with one another's readings. And it was able to send data to the remote online server. What it means is that this node can be used to acquire data from any remote location that has a GPRS network. Therefore, the system, UIoTSn can be applied in many areas of IoT automation for remote data acquisition. Also, the throughput of the system is 95%. The latency of the node was found to be 3 seconds, while the processing time of the node is 135 seconds. The data integrity of the system is 100%. The system proposed in this research will reduce the design to market time for IoT systems to two hours.

References

1. S. Balakrishna, M. Thirumaran, V. K. Solanki, “A Framework for IoT Sensor Data Acquisition and Analysis”, EAI Endorsed Transactions, pp 1-13, 2019. DOI: 10.4108/eai.21-12-2018.159410

2. J. JunHo, B. Jo, K. JungHoon, K. SungJun, W. Han, "Development of an IoT-Based Indoor Air Quality Monitoring Platform". Sensors and Applications in Agricultural and Environmental Monitoring, pp. 1-14, 2020
3. https://www.researchgate.net/publication/338586095_Development_of_an_IoT-Based_Indoor_Air_Quality_Monitoring_Platform
4. E. Balestrieri, L. D. Vito, F. Lamonaca, F. Picariello, S. Rapuano, I. Tudosa, "Research challenges in measurements for Internet of Things systems", Acta IMEKO, Vol.7, No. 4, pp 82-89, 2018
5. https://www.researchgate.net/publication/330284914_Research_challenges_in_Measurement_for_Internet_of_Things_systems
6. V. Vujovic, "Development of a custom Data Acquisition System Based on Internet Of Things", in 2015 International Scientific Conference, UNITECH 2015, pp 339-343, 2015 https://www.researchgate.net/figure/An-Internet-of-Things-based-Data-Acquisition-System_fig2_285345214
7. V. Rao, K. V. Prema, "Internet-of-Things Based Smart Temperature Monitoring System", in 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology, RTEICT 2018, pp 72-77, 2018. DOI: 10.1109/RTEICT42901.2018.9012652
8. A. B. Queyam, R. K. Meena, K. S. Pahuja, D. Singh, "An IoT Based Multi-Parameter Data Acquisition System for Efficient Bio-Telemonitoring of Pregnant Women at Home", in 2018 8th International Conference on Cloud Computing, Data Science & Engineering Noida, 2018. DOI: 10.1109/CONFLUENCE.2018.8442686
9. T. T. Christian, T. Daniel, F. Anaclet, "NodeMCU in Patient's Data Transfer to IoT Platform", Journal of Biomedical Engineering and Medical Imaging, Vol. 5, No. 3, pp 9-18, 2018. DOI: 10.14738/jbemi.53.4358
10. Q. Chi, H. Yan, C. Zhang, Z. Pang, L. D, Xu, "A Reconfigurable Smart Sensor Interface for Industrial", IEEE Transactions on Industrial Informatics, Vol. 10, No. 2, pp 1417-1425, 2014
11. "IoT Data Acquisition System a and Protocol Converters", Scientech Producer, 2020 https://www.scientechworld.com/it-educational-platforms/iot-solutions/iot-data-acquisition-system-and-protocol-convertersGoering [Accessed: June, 4th 2021]
12. W. Kim, H. Ko, H. Yun, J. Sung, S. Kim, J. Nam, "A generic Internet of things (IoT) platform supporting plug-and-play device management based on the semantic web". Journal of Ambient Intelligence and Humanized Computing, 2019 https://link.springer.com/article/10.1007/s12652-019-01464-2 [Accessed: June, 4th 2021]