PAPER • OPEN ACCESS

# Power consumption measurements of WSN based on Arduino

To cite this article: U T Salim *et al* 2021 *IOP Conf. Ser.: Mater. Sci. Eng.* **1152** 012022

View the article online for updates and enhancements.

# Power consumption measurements of WSN based on Arduino

**U T Salim[1], S L Qaddoori[2], A Al-Saegh[1*] and Q I Ali[1]**

[1]Computer Engineering Department, College of Engineering, University of Mosul, Mosul, Iraq.
[2]Electronic Engineering Department, College of Electronics Engineering, University of Ninevah, Mosul, Iraq.

*E-mail: ali.alsaegh@uomosul.edu.iq

**Abstract.** To deal with the deficiencies and limitations for existing wireless sensor network (WSN) nodes, such as low generality, low degree of variability and poor flexibility, Arduino WSN node's development benefits are integrated with Bluetooth wireless communication characteristics. This paper introduces four simple point-to-point WSN power measurement methods. Each WSN node is constructed from Arduino Mega 2560 microcontroller, Bluetooth module, and different sensors. The paper's goal is to characterize the power consumption at different power-saving modes. So, numerous power measurement methodologies are presented to measure the power consumption of the WSN node, in real-time, for different power saving modes such as (idle, sleep, power-down, etc.). These methodologies also determine the power consumption at communication states. Then the results obtained from these states show that the idle mode consumes more power than the other modes. In addition, the transmitted state consumes the major power regardless of the data rate used during the communication between the two ends.

## 1. Introduction
Vast number of projects on wireless sensor networks (WSNs) were carried out duo to the effectiveness of this field of research on the daily life [1,2]. Among the applications of WSNs are environmental data collection[3], robotically handling agriculture irrigation, gathering parameters of the patient's state, identifying mechanical faults, military arena [4] and many others. The main purpose of WSNs is detecting periodically actions within a specific motivating area, process them, and then routing them as digital data to the base station.

Normally, the WSNs comprises of a big number of small sensor nodes that are spatially distributed with limited resources: battery power, capacity of computation, memory and radio range [5]. However, there exists a variety of challenges which may encountered and greatly affect the lifespan of the network; e.g. distance to the base station, distance among nodes, energy consumption[6,7], data processing capability, and may be some others. One of those crucial and dominant issues is the amount of consumed energy which is encountered during routing data among the nodes as well as during the actions of sensing phase [5]. A lot of researches have dealt with energy saving procedures, some of them attempted to achieve that throughout investment different protocols, which comprise MAC protocols, routing protocols, data transmission models, communication architectures, and special protocols, etc. [4]. Other researches attempted to manage energy saving in system level design and the architecture of nodes, whereas the main essential components of a standard node are: Radio (RF) communications, computing unit (MCU/FPGA and Memory), sensor/actuator unit, and power management unit[8].

In order to construct a sophisticated system with low power consumption, a lot of WSN architectures must communicate with very low duty cycle. Each of the nodes within the WSN can be enforced to work with low power, this is achieved by keeping the controller and communication interfaces in a low-power (sleep state) and beside that all the sensors and other peripheral devices are switched off. A node wakes up for a short time periodically, senses if an action is required, circulates information to other nodes within the network then it sleeps again [8]. However, such a system requires some manipulation with the implemented duty cycle in real-time applications, but still these applications will still consume higher power than others.

This paper presents a sophisticated way to measure power consumption of different situations using a number of wireless sensors connected to the Arduino kit. Then, comparison results of those situations are presented and discussed in order to show the power consumption at each state (idle, sleep, send, receive) of WSN operation.

The rest of this paper is introduced as follows: section 2 discusses some of the previous related works. Section 3 describes the used practical electronic circuits, and the way of measuring its power consumption is declared in section 4. Section 5 presents the practical obtained results and gives detail discussion for comparing different work situations. Then the comparison between Arduino Mega and ESP32 is declared in section 6. Finally, section 7 concludes the paper.

## 2. Related Works

To manipulate the power consumption problem and hence prolong the lifespan, many researches have inspected a variety of aspects in measuring and saving power of WSNs.

Due to the different activities of a person, the locations of sensor nodes placed on the body will always change. Thus, using a static routing algorithm causes packet and energy losses during transmission. Khan et al. [9]used a dynamic routing algorithm to address the problem of node movement in wireless body area sensor networks. They used the information of outstanding energies of nodes, hop count to sink distance, and throughput when nodes select the next hop node to forward data.

Elhoseny et al.[10] proposed a self-clustering method using genetic algorithm for improving power consumption in heterogeneous Wireless Sensor Networks. Comparing to other five methods, their proposed one was able to extend the network lifespan. They got improvement of 33.8% and 13%based on the first-node-die and the last-node-die are 33.8% and 13%, respectively, with respect to the second-best performance

Nair et al. [11] proposed a wireless node architecture in order to decrease power consumption in internet of things based wireless sensor networks (IOTWSN). The architecture is based on Bluetooth Low Energy (BLE) wireless standard and also a hybrid topology that connects the nodes to each other. They proved that BLE requires low operating power, able to fast connect, attains data throughput similar to traditional wireless techniques, able to deal with real-time monitoring of data.

Lin et al. [12]presented the Adaptive Transmission Power Control (ATPC) in wireless sensor networks. They showed that those topologies which are based on static power transmission are not suitable for the physical world, instead the adaptable transmission power control has the capability of power maintaining. In ATPC each node figures out a model for each of its neighbours, this model describes the relationship between transmission power and link quality. Then upon that, the individual link quality of a node is dynamically maintained over time.

Deng et al. [13] suggested several variables in sound source localization that affect the path loss exponent. Sound source localization is normally used in battlefield environments where low power consumption is required. They proved that those variables have the ability to reduce power consumption and also reduce localization error.

Pal and Nasipuri[14] proposed Power control and Routing (PCOR) protocol that enforces the sensor nodes in WSN to adjust their energy consumption based on the existing energy resources that permits continuous network operation as long as possible. They achieved that by reducing the overhearing at those nodes suffering from low-energy resources.

Liu et al. [15] used data fusion method in order to decrease the amount of data to be transmitted and hence the amount of consumed energy in WSN. They built their fusion algorithm on hybrid delay-aware clustering (HDC) to achieve a trade-off between the network delay caused by the fusion scheme and the energy consumption.

## 3. Hardware Tools Overview

This section presents some details about the Arduino board and sensors used in our experiments.

### *3.1. Arduino Mega 2560 Board*

The Arduino Mega 2560 is a microcontroller ATmega2560 based board [16,17]. It is an open-source platform allowing the end user to have access to the product design and implementation. The board simply comprises of a microcontroller with all electrical circuits that are necessary for the operation of the microcontroller. The end user can easily upload/download code between the board and a computer throughout a USB cable. Table 1 summarizes the features of the Arduino Mega 2560 board used in our experiments.

The Arduino software comprises of two parts: Arduino Boot-Loader and Arduino IDE. Arduino Boot-Loader is a small program residing within the microcontroller which makes the controller distinct and grants it the power of combination to the Arduino IDE and the Arduino board. The Arduino IDE contains a compiler, serial monitor, etc. Furthermore, Arduino is an open source hardware and software platform which uses Atmel microcontroller as core hardware component and C or C++ as core software language [17].

**Table 1.** Features of Arduino Mega 2560.

| Feature | Type |
|---|---|
| Hardware serial ports | 4 UARTs |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| Digital I/O Pins | 54 (of which 14 provide PWM output) |
| Analog Input Pins | 16 |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 256 KB of which 8 KB used by bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |

### *3.2. Bluetooth*

The HC-05 Bluetooth is a serial communication module that can serves in two working modes: the automatic connection work mode and the order response work mode. One of these modes can be activated via alternating the input voltage to pin (PIO11). When the module works in the automatic connection mode, it has three roles to work on: Master, Slave and Loopback; and the module will transmit and receive data according to the last set role. However, when the module works in the order response mode, the user will have the ability to send AT command in order to set the control parameters. Bluetooth module setup is described in Table 2.

### *3.3. Sensors*

Each sensor differs in its function and has various features so the sensors are uneven in power consumption. In this paper, the sensors have been selected based on the availability in the Lab and ease of connection with the board. Besides, they consume the power from the board without using an external source and give its sensing value instantly that will be used in the communication operations. The used Sensors are: ultrasonic sensor, photoresistor, PIR motion sensor, and temperature and humidity sensor.

**Table 2.** Bluetooth module setup.

| Command | Response | Function |
|---|---|---|
| **Master and Slave module** | | |
| AT | OK | Test |
| AT+ORGL | OK | Restore default status |
| AT+RMAAD | OK | Delete all authenticated devices in the pair list |
| AT+UART= 9600,0,0 | +UART:9600,0,0 OK | Set/ Inquire- serial parameter |
| **Master module** | | |
| AT+NAME= BTMASTER | +NAME:BTMASTER OK | Set/ inquire device's name |
| AT+ROLE=1 | OK | Set/ inquire module role |
| AT+CMODE=0 | OK | Set/ Inquire - connection mode |
| AT+LINK= FCA8,9A,31F7 | OK | Connect device |
| AT+BIND= FCA8,9A,31F7 | OK | Set/Inquire - bind Bluetooth address |
| **Slave module** | | |
| AT+NAME=BTSLAVE | +NAME:BTSLAVE OK | Set/ inquire device's name |
| AT+ROLE=0 | OK | Set/ inquire module role |
| AT+CMODE=1 | OK | Set/ Inquire - connection mode |
| AT+ADDR? | +ADDR:FCA8:9A:31F7 OK | Get module Bluetooth address |

## 4. Experiment Setup

In order to emulate the actual wireless sensor networks, an experiment is implemented using a practical model that is consisting of two electronic circuits. The circuit shown in Figure 1 (its block diagram is in Figure 3) is used for transmitting sensor signals and the other circuit, shown in Figure 2, is for receiving those signals (its block diagram is in Figure 4). The circuits are drawn with the help of the open-source program Fritzing. The experimental model comprises of two Arduino Mega boards that are communicating with each other via two Bluetooth models in the master/slave means. Five different sensors are used for transmitting and receiving signals between the two boards; hence power consumption can be evaluated upon the exchanging of those signals.

The operation at transmitter and receiver sides are declared as flowcharts in Figure 5 and Figure 6. The transmitter checks an interrupt signal for reading the voltage values of sensors if that interrupt is raised, otherwise it goes sleep. All sensor values are assembled is one packet and sent via the Bluetooth. The packet is received at the other side via a Bluetooth as well. A corresponding LED for each of the sensors is lighted depending on the received values of sensors.

### *4.1. Power Management Modes*

The microcontroller on Arduino board supports different modes of operation, these modes make an efficient use of power resources and hence prolong the lifespan. Those modes are activated by software, a brief description for each of them is given here. The idle mode put the CPU in halt state while the timer/counters, SRAM, SPI port, and interrupt system stay in operative state. The power down mode keeps the register contents but freezes the oscillator, deactivating all other chip functions until the following interrupt or hardware reset. The Power-Save mode allows the asynchronous timer to stay working in order to give the user the ability to maintain a timer base while the rest of the device is sleeping [22]. The ADC Noise Reduction (ADCR) mode also put the CPU in halt state and stops the input/output circuitry as well, and leave the asynchronous timer and ADC to be running; this minimizes switching noise during ADC conversions [23]. The standby mode permits almost no time start-up of the device together with very low power consumption, this is achieved by putting all integrated circuits in sleep mode except the crystal/resonator oscillator which stay in running. In Extended Standby mode, both the asynchronous timer and the main oscillator remain to run [22]. Table 3 shows the active clock domains and wake up sources in the different sleep modes.

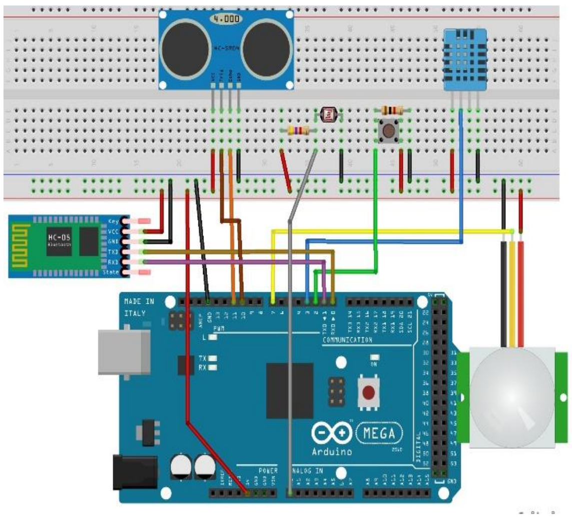

**Figure 1.** Transmitter circuit. Bluetooth as a master with several sensors are connected to the Arduino board. These sensors produce signals to be transmitted to the receiver side.

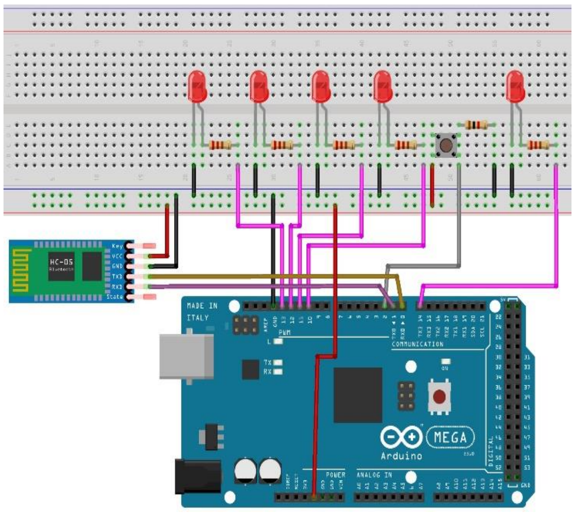

**Figure 2.** Receiver circuit. Bluetooth as a slave with several actuator indicators are connected to the Arduino board. These actuators give indications about the received signals from the transmitter side.

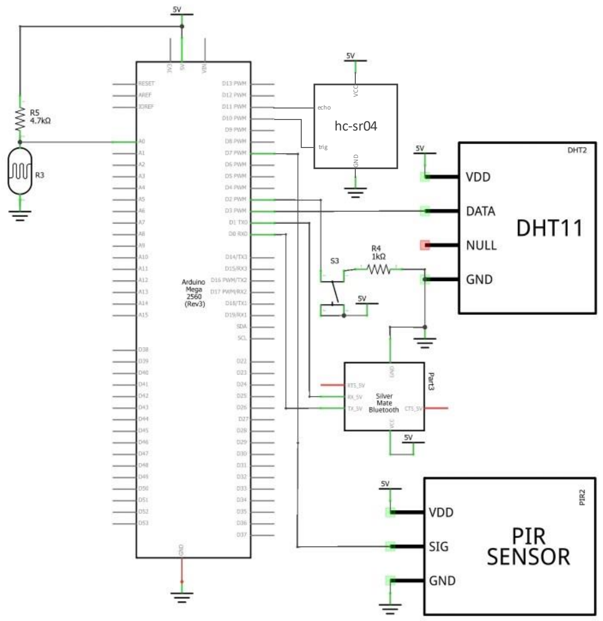

**Figure 3.** Block diagram of the transmitter circuit.

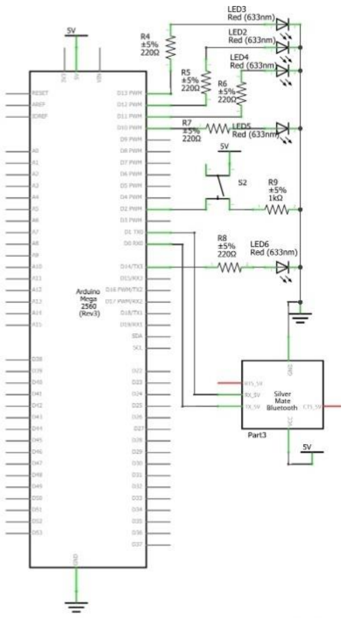

**Figure 4.** Block diagram of the receiver circuit.

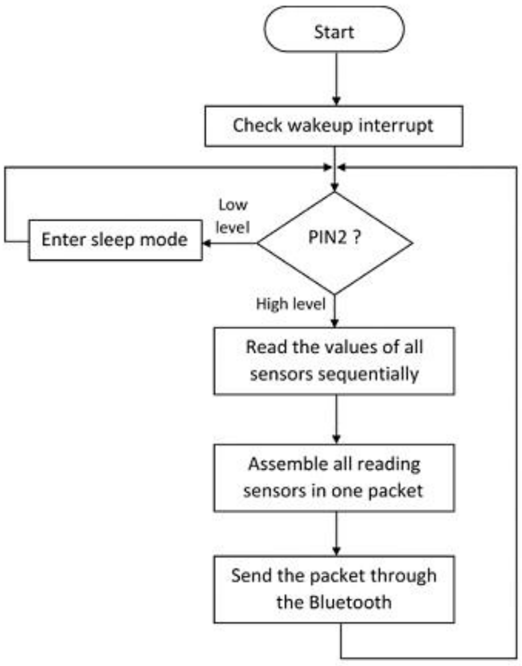
Start
Check wakeup interrupt
Enter sleep mode
Low level
PIN2 ?
High level
Read the values of all sensors sequentially
Assemble all reading sensors in one packet
Send the packet through the Bluetooth

**Figure 5.** Transmitting side flowchart.

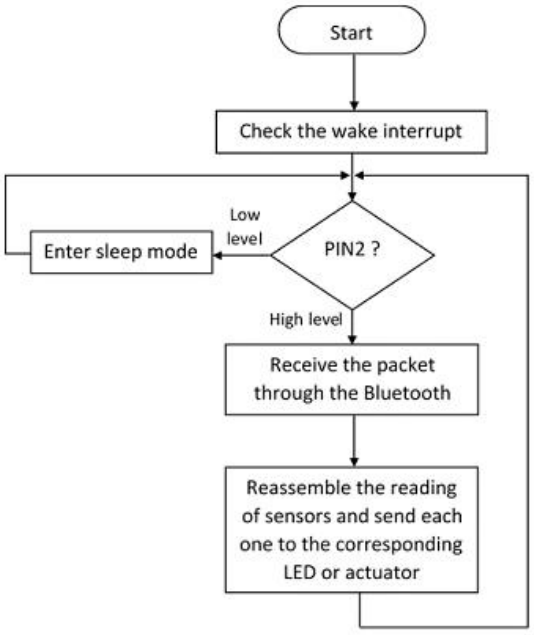
Start
Check the wake interrupt
Enter sleep mode
Low level
PIN2 ?
High level
Receive the packet through the Bluetooth
Reassemble the reading of sensors and send each one to the corresponding LED or actuator

**Figure 6.** Receiving side flowchart.

**Table 3.** Sleep modes of the microcontroller on Arduino board.

| | Active clock domains | | | | | oscillators | | Wake up sources | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sleep mode | Clkcpu | Clk flash | Clk io | Clkadc | Clkasy | Main clock source enable | Timer osc enable | Int7:0 and pin change | Twi address match | Timer2 | Spm\eeprom ready | adc | Wdt interrupt | Other I/O |
| Idle | | | x | x | x | x | x[b] | x | x | x | x | x | x | x |
| ADCNRM | | | | x | x | x | x[b] | x[c] | x | x[b] | x | x | x | |
| Power-Down | | | | | | | | x[c] | x | | | | x | |
| Power-Save | | | | | x | | x[b] | x[c] | x | x | | | x | |
| Standby[a] | | | | | | x | | x[c] | x | | | | x | |
| Extended Standby | | | | | x[b] | x | x[b] | x[c] | x | x | | | x | |

[a] Only recommended with external crystal or resonator selected as clock source.
[b] If Timer/Counter2 is running in asynchronous mode.
[c] For INT7:4, only level interrupt.

### *4.2. Power Measurement Methodologies*

Current and voltage measurements are the most practical methods used to properly calculate the power consumption. Basically, the power value can be calculated using one of the formulas:

$$P = V \times I \tag{1}$$

$$P = \frac{V^2}{R} \tag{2}$$

$$P = I^2 \times R \tag{3}$$

For a known voltage value, the current value can be measured by connecting a personal computer (PC) and Arduino Mega board either through a USB-meter or a Digital Multi-Meter (DMM) as described in Figure 7 and Figure 8. In Figure 7, the electric current value is measured directly through a USB-meter and the measured output is displayed on the USB-meter monitor.

While in Figure 8, the circuit has been splited and the DMM is inserted with the probe which will be put into the current socket and the selector will be pointed on the current choice, then the electric current is passed through the DMM. Where it is measured and displayed on the DMM monitor. Experimental methodologies measurements are presented as illustrated in Figure 9 and Figure 10. In Figure 9, instead of measuring the electric current, the voltage will be measured over 1-ohm resistance using a DMM by putting the probe into the voltage socket and the selector is pointed on the voltage indicator, then the measured output value is displayed on the DMM monitor. Furthermore, Figure 10 is same as Figure 9 but it is used Arduino Uno board instead of the DMM, where the measured output is displayed on a personal computer (PC) through serial monitor Arduino IDE.

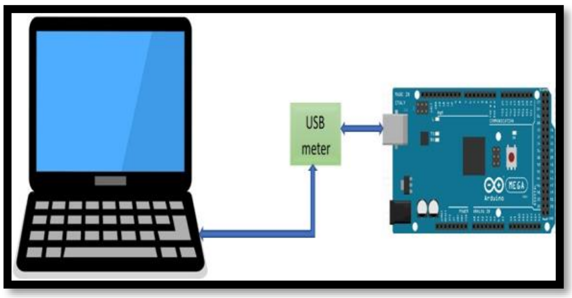

**Figure 7.** The first method (measure the current using a USB meter)

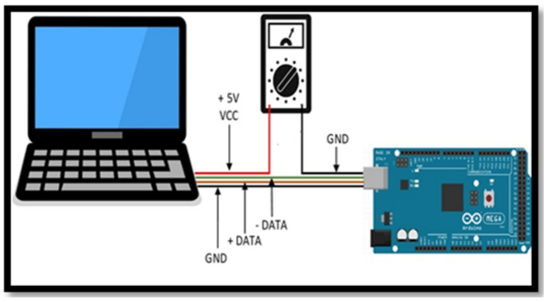

**Figure 8.** The Second method (measure the current using DMM)

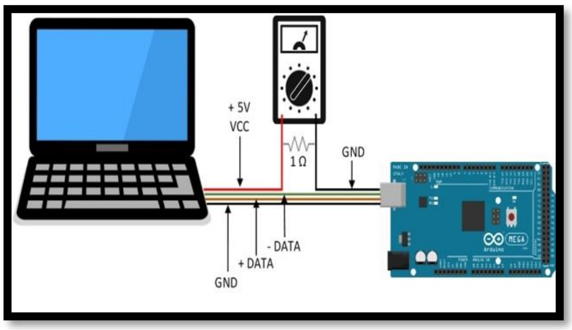

**Figure 9.** The Third method (measure the voltage using DMM)

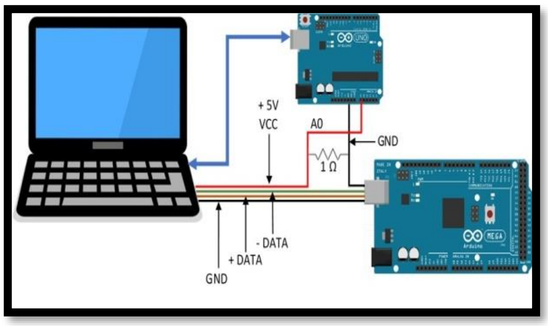

**Figure 10.** The Fourth method (measure the voltage using Arduino-Uno)

## 5. Results and Discussion

Power consumption is tested with respect to the state and role of the device. As, the measuring method checks the impact of sleeping modes on power consumption regardless of the data rate type.

Where, at data rate (9600), the experimental results in Figure 11 show that there is a discrepancy among the measured values using the DMM, USB meter, and Arduino Uno board for the same case. The advantages and disadvantages for each power measuring method are displayed in Table 4.

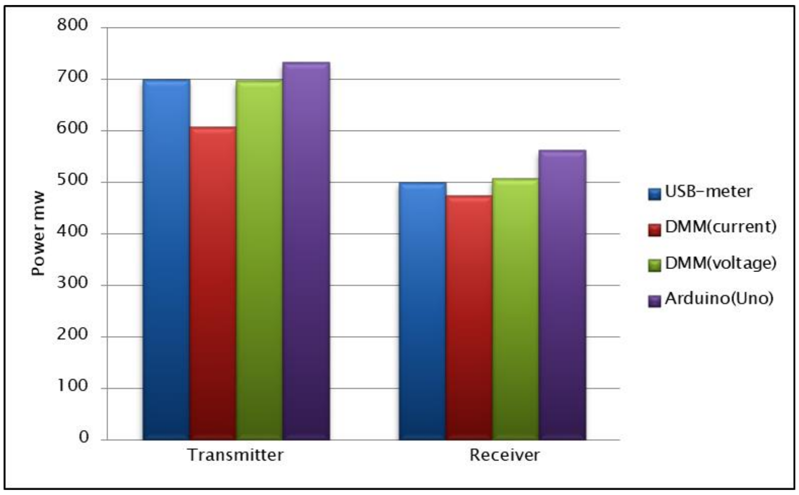

**Figure 11.** Transmitter and receiver power consumption at data rate 9600 for different power measuring methods

**Table 4.** Comparison between the presented power measuring methods.

| Method | Advantages | Disadvantages |
|---|---|---|
| Measure the current using a USB meter | • USB voltage and current can be easily measured.<br>• The USD operation is not influenced, so the current can be measured from any USB device. | • Accurate measurements of current can be obtained when at high current levels but not at low current levels.<br>• It has an internal current shunt resistance of 0.05 ohm, and thus the voltage drop at 2.5A is 0.125V.<br>• Related to the shape of the device the measurement ranges are 3~7.5V and 0~2.5A of the voltage and current, respectively. |
| Measure the current using DMM | • The DMM often has diverse current ranges that are implemented using different shunt resistors to change the value of R. | • It damages the PC USB port if the current draw is too high<br>• The resistance (RA) of the current meter (i.e. Ammeter) must be much smaller compared to the resistance (R) of the test element to have an accurate current measurement. |
| Measure the voltage using DMM | • The DMM often has different voltage ranges that are implemented using different shunt resistors to change the value of R. Some ranges also have a gain stage on the measured voltage. | • The resistance (RV) of the Voltmeter must be much higher compared to the resistance (R) of the test element to have accurate voltage measurement. |
| Measure the voltage using Arduino-Uno | • It is a simple and economic tool that can be used instead of the DMM. | • Its accuracy depends on ADC conversion and is determined by the number of bits and voltage reference. |

Figures 12 and 13 display the monitoring voltage at Transmitter and receiver sides respectively on the serial plotter using Arduino Uno during 5 mins. Each voltage level is multiplied by (5/1024) to find the real voltage value where the value (5) is a source voltage and the value (1024) is ADC resolution (10 bits).

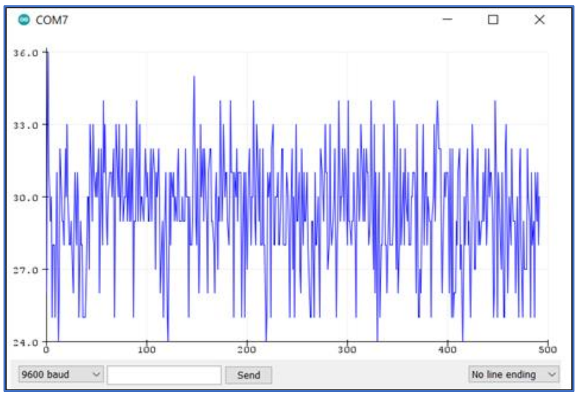

**Figure 12.** Monitoring voltage at Transmitter side using Arduino Uno during 5 mins.

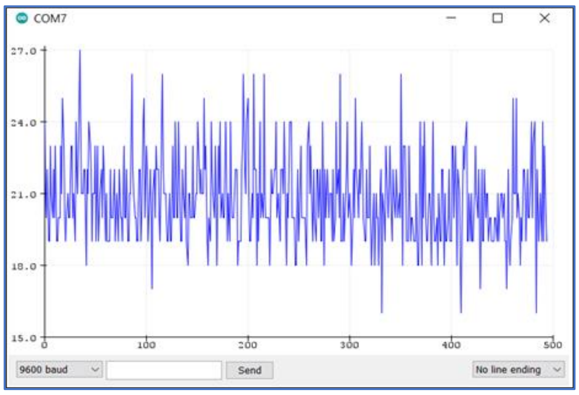

**Figure 13.** Monitoring voltage at receiver side using Arduino Uno during 5 mins.

Figure 14 illustrates the effect of power consumption for different power saving modes. Where, the power consumption of idle mode is larger than other modes, while the PWR_DOWN mode consumes less power, depended on the details in Table 3. This note will be confirmed regardless of the data rate type.

Besides, the different operating modes of the microcontroller unit (MCU) permit power saving based on the application's requirements. Selecting one of the sleep modes causes shutting down unused modules in the in the MCU as stated below for each of the used modes in the experiments result:

Idle mode stops the CPU but allows the SPI, ADC, USART, analog comparator, watchdog,2-wire serial interface, timer/counters, and the interrupt system to continue operating. This sleep mode fundamentally halts the CPU clock and FLASH clock, while permitting the other clocks to work. External and internal interrupts wake up the MCU.

ADC Noise Reduction mode (ADCNRM) stops the CPU but allows the ADC, the external interrupts, timer/counter2, 2-wire serial interface address match, and the watchdog to continue operating (if enabled). This sleep mode principally halts CPU clock, I/O clock, and FLASH clock, while allowing the other clocks to run.

Power-down mode stops the external oscillator, while the 2-wire serial interface, the external interrupts, and the watchdog continue operating (if enabled). Only an external reset, a brown-out reset, a watchdog reset, 2-wire serial interface address match, an external level interrupt on INT7:4, an external interrupt on INT3:0, or a pin change interrupt can wake up the MCU. This sleep mode essentially halts all generated clocks, permitting operation of asynchronous modules only.

Power-Save mode is identical to Power-Down, with one exception: if timer/counter2 is enabled, it will keep running during sleep. This allows the user to maintain a timer base while the rest of the device is sleeping.

Standby mode is identical to the Power-Down mode with the exception that the oscillator is remained running while the rest of the device is sleeping. This permits very fast start-up combined with little power consumption.

Extended Standby mode is identical to the Power-Save mode with the exception that the oscillator is kept running.

The power consumption of different components in presented WSN node are shown in Figure 15. It is indicated that the power consumed by the Board only compared with the power consumed at transmitting and receiving can be insignificant.

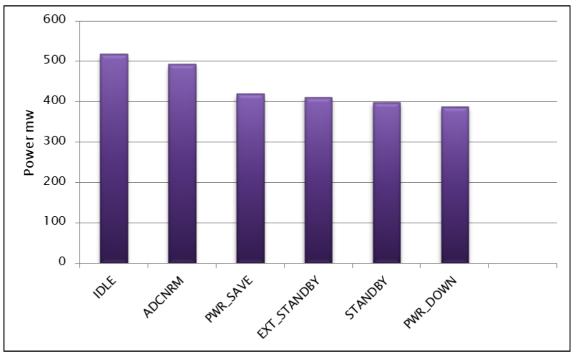

**Figure 14.** Power Consumption at Transmitter Side With Different Sleep Modes at Data Rate 9600.

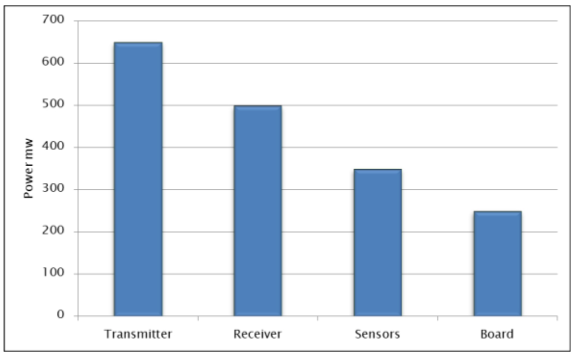

**Figure 15.** Typical power consumption of different components.

## 6. The Arduino Mega compared to the ESP32

These two microcontrollers are totally dissimilar. Where they have diverse hardware architecture such as the processing capability, the memory size, the communications features, the number of GPIOs that they expose, and much more. So, the company that develops and designs the ESP32, has introduced a vast effort in writing software to cover the hardware hole between the Arduino Mega and the ESP32. As a result, the ESP32 operates with the Arduino IDE by the ESP32-Arduino Core installation and the combination between these two microcontrollers is noteworthy [24].

Obviously, the ESP32 has unique features that are not exist in the ArduinoMega, but the Arduino Megais a much better choice because It’s a humbler device, more robust, meeker to program, more forgiving, in addition, to mistakes and problems in the wiring, and easier to set up. In fact, the ESP32 is a better economic option than Arduino Mega, which means that can be gotten a more powerful board for a lesser cost [24]. In the future, the existing Arduino skills can be used to operate with the ESP32 that can be treated as a supercharged ArduinoMega.

## 7. Conclusion

This paper demonstrates several power measurement methods for a WSN node based on the Arduino development board equipped with the Bluetooth module, where both are very prevalent in small-scale research as well as wireless automation solution. These simple, and flexible power measurement methodologies are chosen to calculate how much power is spent at each power-saving mode such as (idle, sleep, power-down, etc.) and communication states. From the testing results, it can be noted that these measurement methods diverse in their accuracy and ease of use and can be used according to the user’s needs. The idle mode consumes power larger than the other modes because most components on the board are turned on, and the major power is consumed at transmitted state regardless of the data rate used during the communication between the two ends. Towards the end, a better function can be attained when integrating the Arduino kit with the Bluetooth module. As the experimental results display the WSN nodes operate stead and the lower power consumption performance is prominent.

## 8. References

[1] Member A S, Karpov E F, Karpova E, Suchkov A, Mironov S, Karelin A, Member A B and Spirjakin D 2015 Compact Low Power Wireless Gas Sensor Node with Thermo Compensation for Ubiquitous Deployment *IEEE Transactions on Industrial Informatics* **11(6)** 1660-1670
[2] Yun W-K and Yoo S-J 2021 Q-Learning-Based Data-Aggregation-Aware Energy-Efficient Routing Protocol for Wireless Sensor Networks *IEEE Access***9** 10737-10750

[3] Javaid N, Cheema S, Akbar M, Alrajeh N, Alabed M S and Guizani N 2017 Balanced Energy Consumption Based Adaptive Routing for IoT Enabling Underwater WSNs *IEEE Access***5** 10040–10051
[4] Peng J and Chen Y 2015 A low energy consumption WSN node *Int. J. Embed. Syst.***7** 318–323
[5] Laouid A, Dahmani A, Bounceur A, Euler R, Lalem F and Tari A 2017 A distributed multi-path routing algorithm to balance energy consumption in wireless sensor networks *Ad Hoc Networks***64** 53–64
[6] Joshi P, Gavel S and Raghuvanshi A S 2021 Estimating Energy Consumption for Various Sensor Node Distributions in Wireless Sensor Networks *Nanoelectron. Circuits Commun. Syst.***692** 289–299
[7] Ali Q I 2018 GVANET project: an efficient deployment of a self-powered, reliable and secured VANET infrastructure *IET Wirel. Sens. Syst.***8** 313–322
[8] Popovici E, Magno M and Marinkovic S 2013 Power management techniques for Wireless Sensor Networks: A review *Proc. 2013 5th IEEE Int. Work. Adv. Sensors Interfaces, IWASI 2013* 194–198
[9] Khan R A, Xin Q and Roshan N 2021 RK-Energy Efficient Routing Protocol for Wireless Body Area Sensor Networks *Wirel. Pers. Commun.***116** 709–721
[10] Elhoseny M, Yuan X, Yu Z, Mao C and El-minir H K 2014 Balancing Energy Consumption in Heterogeneous Wireless Sensor Networks using Genetic Algorithm **7798** 1–4
[11] Nair K, Kulkarni J, Warde M, Dave Z, Rawalgaonkar V, Gore G and Joshi J 2016 Optimizing power consumption in iot based wireless sensor networks using Bluetooth Low Energy *Proceedings of the 2015 International Conference on Green Computing and Internet of Things, ICGCIoT 2015 (IEEE)* 589–593
[12] Lin S 2016 ATPC : Adaptive Transmission Power Control for Wireless Sensor Networks *ACM Trans. Sens. Networks***12** 1–31
[13] Deng F, Guan S, Yue X, Gu X, Chen J, Member S, Lv J and Li J 2017 Energy-Based Sound Source Localization with Low Power Consumption in Wireless Sensor Networks *IEEE Transactions on Industrial Electronics* **64(6)**4894-4902
[14] Pal A and Nasipuri A 2019 Joint power control and routing for rechargeable wireless sensor networks *IEEE Access***7** 123992–124007
[15] Liu X, Zhu R, Anjum A, Wang J, Zhang H and Ma M 2020 Intelligent data fusion algorithm based on hybrid delay-aware adaptive clustering in wireless sensor networks *Futur. Gener. Comput. Syst.***104** 1–14
[16] Kumar N S, Vuayalakshmi B, Prarthana R J and Shankar A 2017 IOT based smart garbage alert system using Arduino UNO *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON* 1028–1034
[17] Badamasi Y A 2014 The working principle of an Arduino *2014 11th International Conference on Electronics, Computer and Computation (ICECCO)* 1–4
[18] Li S E, Li G, Yu J, Liu C, Cheng B, Wang J and Li K 2018 Kalman filter-based tracking of moving objects using linear ultrasonic sensor array for road vehicles *Mech. Syst. Signal Process.***98** 173–189
[19] Qi Y, Soh C B, Gunawan E and Low K S 2015 Ambulatory Measurement of Three-Dimensional Foot Displacement during Treadmill Walking Using Wearable Wireless Ultrasonic Sensor Network *IEEE J. Biomed. Heal. Informatics***19** 446–452
[20] Amri I, Dian Atmajati E, Salam R A, Yuliza E, Munir M M and Khairurrijal 2017 Potentiometer a simple light dependent resistor-based digital *Proceeding - 2016 Int. Semin. Sensors, Instrumentation, Meas. Metrol. ISSIMM 2016* 24–27
[21] Yang W, Qiao S, Song Q, Liu Z and Yang J 2015 The design and implementation of wireless temperature and humidity control system based on nRF905 *Proc. 2015 10th IEEE Conf. Ind. Electron. Appl. ICIEA 2015* 753–756
[22] Berthier F, Beigne E, Vivet P and Sentieys O 2015 Power gain estimation of an event-driven wake-up controller dedicated to WSN's microcontroller *Conf. Proc. - 13th IEEE Int. NEW*

*Circuits Syst. Conf. NEWCAS 2015* 1–4

[23] Wannenburg J and Malekian R 2015 Body Sensor Network for Mobile Health Monitoring, a Diagnosis and Anticipating System *IEEE Sens. J.***15** 6839–6852

[24] Maier A, Sharp A and Vagapov Y 2017 Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things Proceeding - *2017 Internet Technologies and Applications ITA* 143-148