Analysis of Wireless Sensor Node Based on Arduino In Home Automation

K.Ravi Kumar[1], N. Sivaiah[2]
[1,2]Assistant Professor, Dept of ECE
[1, 2]VasireddyVenkatadri Institute of Technology, Nambur, AP,India

Abstract- *Internet of Things (IoT) is an extension of the Internet, which now includes physical objects of the real world. The main purpose of Internet of Things is to increase a quality of people's daily life. A smart home is one of the promising areas in the Internet of Things which increases rapidly. It allows users to control their home devices anytime from any location in the world using Internet connectivity and automate their work based on the physical environment conditions and user preferences. The main issues in deploying the architecture of IoT are the security of the communication between constrained low-power devices in the home network and device performance. These issues have been deepened with the spread of cheap and easy to use microcontrollers which are used by electronic enthusiasts to build their own home automation projects. In this paper we proposed the correlation between security, power consumption of constrained IoT device, and performance of wireless communication based on a model of a home automation system with a sensor node. Sensor node was implemented using Arduino Nano microcontroller and RF 433 MHz wireless communication module.*

Keywords- Arduino microcontroller, home automation system, Internet of Things, power consumption.

I. INTRODUCTION

The main aim of Internet of Things concept is to make different small devices able to communicate to each other and with the user to collect and exchange data over the Internet. In other words, Internet of Things (IoT) is an extension of the Internet, which now includes physical objects of the real world. When IoT is expanded with sensors and actuators, it becomes a part of the more general class of cyber-physical systems. Data processing helps us to automate some actions and add more intelligence to our system. The main purpose of Internet of Things is to increase a quality of people's daily life [1]. In today's world Internet of Things applications cover different areas from business to human dwellings. We can highlight the following domains: transportation and logistics, health-care, smart environment (city, office, and plant), industrial, and consumer [2, 3].

Some of the IoT devices have constrained capabilities and limited access to power. Constraints of the devices affect the characteristics of the communication. Battery capacity also limits the performance of the wireless communication channel and lifetime of the device. Arduino microcontrollers are most popular for controlling sensors and actuators in a physical world. But they are very cheap, simple and energy-intensive constrained devices with low memory and computational capabilities. The main aims of this paper is to determine how security affects the power consumption of a constrained IoT device and performance of the wireless communication channel based on Arduino Nano microcontroller and RF 433 MHz communication module in home automation scenario.

II. EXISTING WORK

There are many works that show a comprehensive survey of the Internet of Things and smart home applications. This topic is quite popular among the researchers because of its actuality. For example, L. Atzori et al. in the paper [4] give an overview of different visions of Internet of Things paradigm and their technologies with an emphasis on using RFID technology as a basis of IoT. Also, open issues that require further research are provided. In papers [5, 6] open issues in Internet of Things and security aspects, in particular, are considered. J. Granjal et al. [6] deeply analyze existing protocols and mechanisms used in IoT and how they ensure fundamental security parameters of communication. The same document also creates a list of proposals and alternative approaches from different researchers. These papers can be used to concentrate on current research in IoT security. The best of our knowledge there is no work that would give answers to all defined in this paper research questions. But some works listed above were quite helpful for this research. For example, our experiments were based on methodology given in [13] with using simple laboratory equipment like in [10] and easy-to-use Arduino electronics platform. Power saving techniques described in 11, 12] help to understand the potential for improving the energy efficiency of Arduino microcontrollers and identify future work. For theoretical analysis of protocols and mechanisms in IoT, open issues in IoT papers [8] give the good background knowledge. For building a model of a smart home, defining its use cases and

requirements works [9] were useful. Smart home model and model of a capillary network with heterogeneous devices including unidirectional and non-IP devices defined in [9] inspired to create a model of a home automation system with a sensor node.

III. SMART HOME AND HOME AUTOMATION SYSTEM

As it was mentioned, the major part of IoT devices concentrates in consumer field what covers: wearables, media devices, home automation, and smart appliances [3]. The coverage of these applications is usually limited by Local Area Network (LAN), Personal Area Network (PAN), or Body Area Network (BAN) inside consumer's home. The home technology is developing rapidly to integrate all home systems under one centralized management system which forms Home Area Network (HAN). But problem complexity and incompatibility of multiple technologies, products from different vendors result in a high price of such system and limit its practical implementation. The main components of this structure are: home automation system (e.g., lighting, HVAC (heating, ventilation and air conditioning), security locks); energy management system or smart energy (smart appliances, e.g., heating/AC, water heater, oven, dishwasher, dryer, plug-in electrical car); entertainment devices (smartphones, laptops, PCs, tablets, TVs, home cinema systems, game consoles, etc.) in PAN; wearables (e.g., smartwatches, heart rate monitors, body temperature sensors, accelerometers, blood pressure sensors) in BAN. Smart home can include a lot of different systems and applications, as illustrated in Figure.

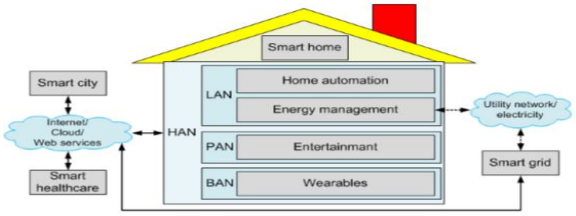

Fig 1: Structure of smart home.

IV. METHODOLOGY

In this paper we use Arduino Nano with RF transmitter module as a sensor node and Arduino Mega with RF receiver as a control node in HAN. Arduino Nano consumes less power, it is compact and we doesn't need so many analog and digital pins, just a few for sensor readings and wireless communication.

RF 433 MHz wireless communication modules

There are a lot of low-cost RF transmitter and receiver modules on the market. They are found in different shapes, but functionally they are the same. Radio modules FS1000A and MXRM-5V work on unlicensed frequency 433.92 MHz in LPD433 (Low Power Device) band. They can be connected to the Arduino microcontroller with just three pins. They are usually used in remote control equipment and home security products. First one, FS1000A, is a transmitter, the second, MX-RM-5V, is a receiver. Transmitter uses digital input to transmit signal with ASK (Amplitude Shift Keying) modulation to a distance of 100 meters within line of sight (no external antenna). The communication range can be increased by connecting an antenna to the transmitter and receiver. Transfer rate is about 4 Kb/s. The receiver includes AGC (Automatic Gain Control) which increases the range of reception but may cause chaotic false signals on the receiver. It is very sensitive to ripple on the power line.In this we are going to use Transmitter FS1000A and Receiver MX-RM-5V.

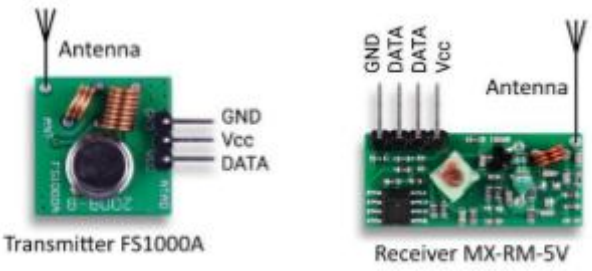

Fig 2: Transmitter and Receiver

There are a few libraries for Arduino to help using 433 MHz RF modules (VirtualWire, RadioHead, RC-Switch, RemoteSwitch, iarduino_RF433) but it's optional to use them. Some of them are using hardware timers, some of them using external interrupts. There are some advantages and disadvantages in both of them, but we will need external interrupts for sleep control. We will use VirtualWire[14] library that provides features to send short broadcast messages. Maximum payload size is 27 bytes. It takes care about sync patterns, data formatting, a balance of 0 and 1 bits, and error checking allowing for best performance from cheap radio circuits. Messages are sent with 4-to-6 bit encoding without retransmitting or acknowledgment. VirtualWire uses Timer1 of the Arduino, so it can affect capabilities of some PWM pins [14].

Design of experiment

Let's suggest we have a sensor node based on Arduino Nano microcontroller and we want to send periodically information from it to the coordinator based on Arduino Mega to process and manage the cooling and heating devices and adjust the light in the room. One or few sensors

can be connected to our microcontroller and it will use wireless radio for data transmission as depicted in Figure 3.

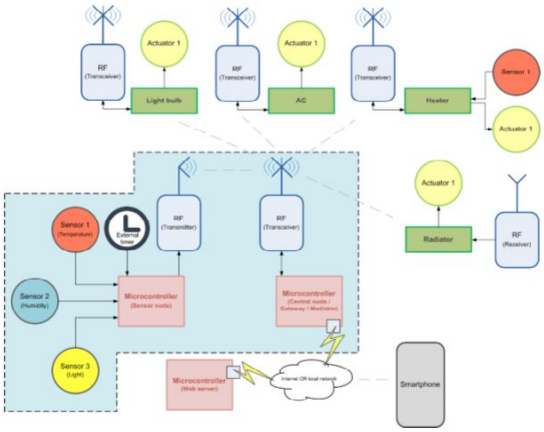

Fig 3: Home automation model with a sensor node

The main aim is to save power which is the most precious resource of the battery-powered device. The power consumption of the device during sleep mode is much less than during active stage, so increasing the time interval between activities will increase the lifetime of the device. It will take less time for a processor with higher frequency to perform all the calculations, decreasing the active time, but, probably it will consume much energy. Wireless communication will determine a lot of performance characteristics of the channel like bandwidth, data rate, and throughput, max range, a power consumption of RF transmitter, and so on.

Let's simplify the model and assume, that there are only a few stages of activity of the node: sleep, reading the sensor data, package formation, data encryption (optional), sending the package, when done go to the sleep mode again. We will assume that data from even a few sensors usually will not exceed the size of the package and since the minimization of package size is desirable we will not consider larger data sizes.

V. EXPERIMENT

To be able to measure the power consumption of the Arduino Nano microcontroller during the lifetime cycle we need to set up it for sleeping. There are four mechanisms for waking the Arduino from sleep:

1. using an internal timer;
2. using the watchdog timer;
3. using the hardware UART (serial interface over USB);
4. using an external interrupt (change in pin state).

When using internal timers (Timer0, Timer1, or Timer2) Arduino will be woken up periodically. In that case, there are some limitations on maximum sleep interval that depends on the clock size and maximum allowing sleep mode. The maximum timeout period is only 16.4 ms. Using that mechanism can be irrationally for real world sensor nodes applications because of the power loss during the constant waking. But they can be used when we should constantly monitor the situation. The maximum allowed sleep mode is only SLEEP_MODE_PWR_SAVE for Timer2. In contrast, the watchdog timer provides the lowest sleep mode and timeout period up to 8 s.

Using serial interface or external interrupt will allow us to wake it up at any time. The control can be done with an external timer or event-based sensor. We can also choose any required sleep mode. Arduino Nano has two external interrupt pins available and we will use pin 3 to control the Arduino Nano sleep modes and synchronize it during the measurements. Also, we connect data pin of our RF 433 MHz FS1000A transmitter to the pin 2 of Arduino Nano for communication as shown in Figure 4.

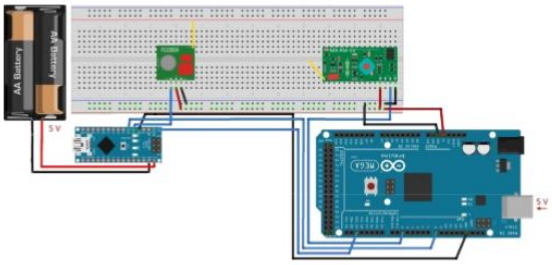

Fig 4: Setup of experiment

Measuring the current consumption of different parts of the program during the experiment 3a using a microammeter allowed summarizing results for data transmission in Table 5.1. In this table, we will denote with symbol minus ("-") measurement taken without feeding the RF module from the Arduino (or the same power supply as a microcontroller) and otherwise with symbol plus ("+"). RF transmitter can be powered by 3.3 or 5 V from the Arduino. The experiment showed 19.5 A of measurement value on the microammeter during the encryption regardless of the key size, that is about 9.95 mA. Other parts of the program showed 20 A for delay function, random function, and package formation part which is equivalent to 10.2 mA. We will consider that value as a reference value. From the table, we can say that current consumption of data transmission doesn't depend on the speed of the communication and it's almost the same for sending 4 bytes or 16 bytes (encrypted data). RF transmitter itself consumes about 5 mA. The difference in current depending on transmitter power is about 1 mA. As you

can see data transmission in "3.3+" mode consumes almost 14.3 mA. Table 1 shows results for current measurements in different sleep modes of the Arduino Nano. As you can see, for example, in PWR_DOWN sleep mode Arduino consumes about 3.6 mA.

		Measurement value, μA				Current, mA			
Sleep mode	*Level*	-	*LED is on*	*3.3+*	*5+*	-	*LED is on*	*3.3+*	*5+*
SLEEP_MODE_PWR_DOWN	0	7	7	7	7	3.57	3.57	3.57	3.57
SLEEP_MODE_STANDBY	1	7	8	8	8	3.57	4.08	4.08	4.08
SLEEP_MODE_PWR_SAVE	2	8	8	8	8	4.08	4.08	4.08	4.08
SLEEP_MODE_EXT_STANDBY	3	8	8	8	8	4.08	4.08	4.08	4.08
SLEEP_MODE_ADC	4	9	9	9.5	10	4.59	4.59	4.85	5.1
SLEEP_MODE_IDLE	5	18	21	22	24	9.18	10.71	11.22	12.24

Table 1: Current consumption during the sleeping at different sleep modes

VI. RESULTS

We can see the comparison of measurement results using two different methods for different sleep modes.

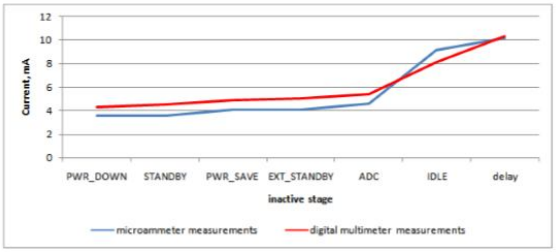

Fig 5: Comparison of measurement results of current for different sleep modes

when we send data with low periodicity, roughly speaking, communication speed and encryption don't influence the lifetime of the battery. If we will use external interrupt sleep management with the highest sleep mode PWR_DOWN of Arduino Nano in our home automation project for a sensor node to send 4 bytes of data every 5 minutes with RF 433 MHz FS1000A transmitter battery of a sensor node will serve 16.8 days instead of 6 days when no sleep mode is used. Values include powering the RF transmitter from 3.3 V pin of the same microcontroller ("3.3 +" mode). Is worth mentioning that no other extreme power saving techniques like cutting of the internal LED or disabling a brown-out detection were investigated in this research. For example, in [11, 12] authors consider a various general power saving techniques for microprocessors and in particular for ATmega328. The only additional thing we have done that we set all unused pins of Arduino as inputs and enabled pull-up resistors on them [16]. Theoretically, applying various other techniques can reduce the power consumption of ATmega328 processor in power-down sleep mode to insignificant values lower than self-discharge of the battery. That will allow the lifetime of years for battery powered nodes.

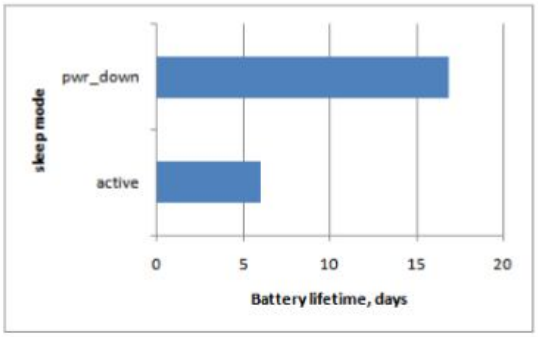

Fig 6: Battery lifetime for 5 minutes period

The battery lifetime can be calculated using equation below based on a formula from [15].

$$Battery\ life\ (hours) = \frac{Battery\ capacity\ (mAh)}{\frac{I_{active}(mA) \cdot t_{active}(ms) + I_{sleep}(mA) \cdot t_{sleep}(ms)}{t_{active}(ms) + t_{sleep}(ms)}}$$

we can conclude that when our sensor node is sending data frequently using a low speed of communication with encryption the lifetime of the battery will be almost the same no matter if we use sleep mode or not. The higher speed, the longer battery life. The best characteristics are when the data rate of 6000 bps and power-down sleep mode are used. In that case, when using AES encryption the lifetime of the battery will be 11.5 days and 13.7 days with no encryption. As you probably remember from Table 1 current consumption of the device is essentially independent of the transmission speed. However, the maximum speed is limited because of the decrease in performance of the communication channel (maximum range and packet loss). On the speed of 6000 bps RF 433 MHz modules still operate correctly

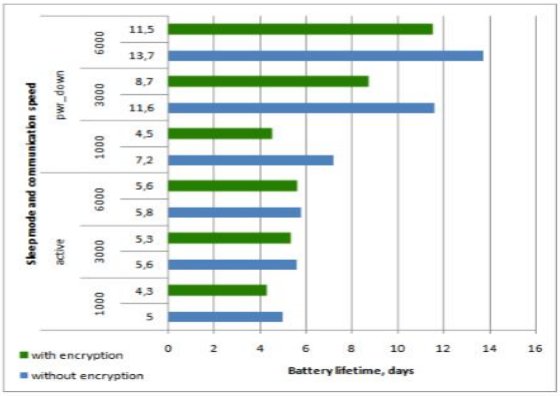

Fig 7:Battery lifetime for a period of 300 ms depending on encryption, sleep management, and communication speed

VII. CONCLUSION

In this paper, a model of a home automation system with a sensor node based on Arduino Nano was described. A

performance of the device was evaluated by measuring execution time and power consumption when using different communication and security parameters and different power supply strategies. Sleep management with an external interrupt allowed us to reduce the power consumption of the device and synchronize it during the measurements. An expected lifetime of the battery was calculated. After analyzing the results we have formulated some practical recommendations for developers of smart home applications in order to increase the lifetime of the device, security, and performance of the communication. The results in this paper showed that using sleep modes for Arduino does not have so much effect on the battery lifetime as it was expected because of the extra hardware of the board.

REFERENCES

[1] V. Namirimu, "User requirements for internet of things (iot) applications – an observational study", Master of Science in Software Engineering, Blekinge Institute of Technology, Karlskrona, Sweden, 2015.

[2] M. Razzaque, M. Milojevic-Jevric, A. Palade and S. Clarke, "Middleware for Internet of Things: A Survey", IEEE Internet of Things Journal, vol. 3, no. 1, pp. 70-95, 2016.

[3] P. Smutný, "Different perspectives on classification of the Internet of Things", in Carpathian Control Conference (ICCC), 17th International, 2016, pp. 692-696.

[4] L. Atzori, A. Iera and G. Morabito, "The Internet of Things: A survey", Computer Networks and ISDN Systems, vol. 54, no. 15, pp. 2787-2805, 2010.

[5] K. Gupts and S. Shukla, "Internet of Things: Security challenges for next generation networks", in International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH), 2016, pp. 315 - 318.

[6] J. Granjal, E. Monteiro and J. Sa Silva, "Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues", IEEE Communications Surveys & Tutorials, vol. 17, no. 3, pp. 1294-1312, 2015.

[7] R. Giuliano, F. Mazzenga, A. Neri and A. Vegni, "Security Access Protocols in IoT Capillary Networks", IEEE Internet of Things Journal, pp. 1-12, 2016.

[8] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities", IEEE Internet of Things Journal, vol. 1, no. 1, pp. 22-32, 2014.

[9] C. Lee, L. Zappaterra, K. Choi and H. Choi, "Securing Smart Home: Technologies, Security Challenges, and Security Requirements", in Workshop on Security and Privacy in Machine-to-Machine Communications (M2MSec'14), 2014, pp. 67-72.

[10] C. Trasviña-Moreno, R. Blasco, R. Casas and A. Marco, "Autonomous WiFi Sensor for Heating Systems in the Internet of Things", Journal of Sensors, vol. 2016, pp. 1-14, 2016.

[11] N. Gammon, "Gammon Forum: Electronics: Microprocessors: Power saving techniques for microprocessors", Gammon.com.au, 2012. [Online]. Available: http://www.gammon.com.au/forum/?id=11497. [Accessed: 17- May- 2017].

[12] "Reducing Arduino Power Consumption", Learn.sparkfun.com, 2016. [Online]. Available: https://learn.sparkfun.com/tutorials/reducing-arduino-power-consumption. [Accessed: 02- May- 2017].

[13] M. Levy, "Understanding the Real Energy Consumption of Embedded Microcontrollers", Digikey.com, 2012. [Online]. Available: https://www.digikey.com/en/articles/techzone/2012/jun/understanding-the-real-energy-consumption-of-embeddedmicrocontrollers. [Accessed: 02- May- 2017].

[14] M. McCauley, VirtualWire. Documentation for the VirtualWire communications library for Arduino, 1st ed. 2013, pp. 1-9.

[15] Humidity and Temperature Sensor Node for Star Networks Enabling 10+ Year Coin Cell Battery Life, 2nd ed. Texas Instruments, 2016, pp. 1-35.

[16] "Hush little microprocessor... AVR and Arduino sleep mode basics", Engblaze.com. [Online].Available: http://www.engblaze.com/hush-little-microprocessor-avr-and-arduinosleep-mode-basics/. [Accessed: 01- May- 2017].