Master of Science in Electrical Engineering
with emphasis on Telecommunication Systems
June 2017


# Comparison of Wireless Communication Technologies used in a Smart Home:

## Analysis of wireless sensor node based on Arduino in home automation scenario

**Oleh Horyachyy**

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering with emphasis on Telecommunication Systems. The thesis is equivalent to 20 weeks of full time studies.

**Contact Information:**
Author(s):
Oleh Horyachyy
E-mail: olho17@student.bth.se

University advisor:
Prof. Kurt Tutschku
Department of Computer Science and Engineering

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

# Abstract

**Context.** 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. Battery lifetime is a key QoS parameter of a battery-powered IoT device which limits the level of security and affects the performance of the communication. 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.

**Objectives**. In this study, we investigated wireless communication technologies used in low-power and low-bandwidth home area networks to determine which of them are most suitable for smart home applications. We also investigated 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.

**Methods**. To achieve the stated objectives of this research following methods were chosen: literature review to define common applications and communication technologies used in a smart home scenario and their requirements, comparison of wireless communication technologies in smart home, study of Arduino microcontroller technology, design and simulation of a part of home automation project based on Arduino, experimental measurements of execution time and power consumption of Arduino microcontroller with RF 433 MHz wireless module when transmitting data with different levels of security, and analysis of experimental results.

**Results**. In this research, we presented a detailed comparison of ZigBee, WiFi, Bluetooth, Z-Wave, and ANT communication technologies used in a smart home in terms of the main characteristics. Furthermore, we considered performance, power consumption, and security. A model of a home automation system with a sensor node based on Arduino Nano was described with sleep management and performance evaluation. The results show that the battery lifetime of Arduino in a battery-powered sensor node scenario is determined by the communication speed, sleep management, and affected by encryption.

**Conclusions**. The advanced communication strategy can be used to minimize the power consumption of the device and increase the efficiency of the communication. In that case, our security measures will reduce the productivity and lifetime of the sensor node not significantly. It's also possible to use symmetric encryption with smaller block size.



**Keywords:** Arduino microcontroller, home automation system, Internet of Things, power consumption

# Acknowledgements

I would like to express my heartfelt gratitude to my supervisor Prof. Dr. Kurt Tutschku for the guidance, patience and continuous support throughout my thesis. I would like to extend my sincerest thanks and appreciation to Anders Carlsson for his continuous support, constant and timely advice and assistance. It wouldn't have been possible to finish this thesis without their support, motivation, and encouragement.

I would like also thank Maryna Yevdokymenko, Helen Tkachova, Stepan Voytusik, Volodymyr Sokolov, and Oksana Yevsieieva for their motivation and encouragement, pieces of advice, and useful feedback.

I would like to thank my wonderful family and friends for their emotional support, prayers, and constant encouragement throughout my entire program.

# List of Figures

# List of Tables

# List of Abbreviations

4G…… 4th Generation of mobile telephony
6LoWPAN... IPv6 over Low power WPAN
AC…….....Air Conditioner
ACL……. Access Control List
ADC…….Analog-to-Digital Converter
AES……..Advanced Encryption Standard
AGC…….Automatic Gain Control
ASK……. Amplitude Shift Keying
BAN…….Body Area Network
BLE……..Bluetooth Low Energy
CBC……. Cipher Block Chaining
CCM…… Counter with CBC-MAC
CoAP…….. Constrained Application Protocol
CPU……. Central Processor Unit
CRC……. Cyclic Redundancy Check
CSMA/CA... Carrier Sense Multiple Access with Collision Avoidance
CTR……. Counter
DC……Direct Current
DECT ULE.. Digital Enhanced Cordless Telecommunication Ultra Low Energy
DES……..Data Encryption Standard
DIY…….. Do-It-Yourself
DNS……. Domain Name System
DSSS……....Direct Sequence Spread Spectrum
EAP……..Extensible Authentication Protocol
ECC……. Elliptic Curve Cryptography
EEPROM….Electrically Erasable Programmable ROM
EXI…….. Efficient XML Interchange
FFD……..Full-Function Device
FHSS…….Frequency Hopping Spread Spectrum
GPS……..Global Positioning System
GTS……..Guaranteed Time Slot
HAN…….Home Area Network
HTML……..Hypertext Markup Language
HTTP……... Hypertext Transfer Protocol
HVAC……..Heating, Ventilation and Air Conditioning
IDE…….. Integrated Development Environment
IEEE…….Institute of Electrical and Electronics Engineers
IP……..Internet Protocol
IPsec…….IP security
ISM…….. Industrial, Scientific and Medical
LAN…….Local Area Network
M2M…… Machine-to-Machine
MAC…… Message Authentication Code
MAN……Metropolitan Area Network
NFC……. Near Field Communication
PAN……. Personal Area Network
PANA…….. Protocol for carrying Authentication for Network Access
PC…… Personal Computer
PIN……...Personal Identification Number
PLC……. Power Line Communication
QoS…….. Quality of Service
IoT……... Internet of Things
RAM…… Random Access Memory
REST……... Representational State Transfer
RF…… Radio Frequency
RFC……..Request for Comments
RFD……. Reduced-Function Device
RFID…… RF Identification
ROM…… Read Only Memory
RPL……..Routing Protocol for Low power and Lossy Networks
RTS/CTS…. Request to Send/Clear to Send
SCADA……Supervisory Control and Data Acquisition
SMS……. Short Message Service
TCP……..Transmission Control Protocol
TEA……. Tiny Encryption Algorithm

TLS…….. Transport Layer Security
TV…… Television
UART…….. Universal Asynchronous Receiver/Transmitter
UDP……. User Datagram Protocol
USB……. Universal Serial Bus
WAN……Wide Area Network
WEP…….Wireless Equivalent Privacy
WPA…… Wireless Protected Access
WPAN……. Wireless Personal Area Network
WSAN……. Wireless Sensor and Actuator Network
WSDL……..Web Services Description Language
WSN…… Wireless Sensor Network
XML…… Extensible Markup Language

# Contents

# 1 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. Sensors help us to get more precise information about our world. When combined with actuators they give us more control over our environment. 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, plant), industrial, and consumer [2, 3]. A lot of small sensors and smart devices are scattered all over our cities, houses of our neighbors, and even we always have at least one of those devices near us. Yes, I am talking about our smartphones, laptops, and PCs. You can even have a smartwatch on your hand. But if we want to get more intelligence, usability, and functionality from our smart devices we should add some communication capabilities. For an example, our smartwatch with heart rate monitoring feature can send this data wirelessly to our smartphone using Bluetooth (or Bluetooth Low Energy), to our PC using home WiFi or directly to the cloud. Our smartphone can have application for monitoring that data and we can share it with our doctor or fitness trainer using WiFi at home or 4G outdoor. Usually, smartwatch works very closely with a smartphone, paired with it, it may function as a remote to the phone. It can play music, display text messages like SMS and emails, and show GPS location.

Another promising area in the Internet of Things is smart home. Our home becomes a part of IoT as soon as we have two devices communicating inside. A variety of communication technologies and their combinations is used for different applications in the smart home. So, it is important to know their characteristics to select one most suitable for a particular application. They are used to interconnect heterogeneous devices inside the house, mostly wirelessly. Smart home concept includes two major components: home automation and smart energy. Home automation involves control and automation of lighting, heating, conditioning, and home security. As an example, thermostat inside the room with a built-in temperature sensor can send data to the HVAC (heating, ventilation and air conditioning) system using ZigBee to adjust the temperature. Similarly with light. Motion detection sensor can send information to the smart bulb to turn on/off light or adjust its brightness depending on the presence of people in the room. A glass break detector listens for the pattern of breaking glass and can send wired or wireless using DECT Low Energy alert signal. Home automation system also involves remote monitoring and controlling features. Using your smartphone or PC you can monitor sensors data and control your smart devices. Usually, it involves using home WiFi and requires the use of additional equipment. Smart energy is used to monitor and control the use of electricity of home appliances. For example, a smart dishwasher can get

information from your home smart meter about electricity pricing using ZigBee (or ZigBee IP) and delay or adjust its work. The smart appliances can be controlled and scheduled over the web or even a TV.

Some of the IoT devices have constrained capabilities and limited access to power. As a part of a home automation system, there are a lot of simple battery-powered sensors and actuators like temperature and humidity sensors, light sensors, motion detectors, fire alarms, etc. They are an important part of the home automation system and should work as long as possible without the need of changing the battery. It can be very annoying for the end user and even difficult to have a physical access and periodically change batteries on all of them. 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.

Security of the communication between devices is important because there is a lot of personal information spreading in our houses. Even safety of our family can depend on privacy, integrity, and availability of this information. In practice, all the security of our home network may rely on one shared network key. Sometimes vendors use one master key for all their devices. And we should remember that the weakest node in the system will define the level of security. An attacker can easily get control of our home and make us victims of our own technologies. That's why security is an important requirement for the smart home. But security requires extra power and computation capabilities. Not all devices in our home can provide a high level of security, for an example, simple devices like a temperature sensor or a light bulb.

## 1.1 Problem statement

It should also be noted that for now, smart devices are at least ten times more expensive than usual ones, for an example, smart bulbs. So, many customers are not willing to buy them. Therefore reduction of cost and power consumption of IoT devices is an important issue. And with the great popularity of cheap and easy programmable microcontrollers many people do their own home automation projects. A good example of do-it-yourself (DIY) electronics is Arduino and Raspberry Pi microcontrollers. As an example, many cases of using these microcontrollers for different monitoring and control applications in home automation are shown in [4] and [5]. 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. And in that situation, a question of security is a topical issue. It's hard to implement heavy cryptography algorithms on them like asymmetric cryptography and a lot of simple communication modules for Arduino don't use any security protection at all. If you want to add some security features to your home automation project you can implement that security algorithm by yourself or use an existing library for Arduino. But there is a question: What level of security does your project need and if it's possible to run it on your constrained device? Fortunately, most home automation applications don't require high performance and security, so we can save power, what is very important for battery-powered devices. But, from the other side, we should not sacrifice safety for performance and usability.

Constrained IoT devices communicate to each other in capillary networks using low-power and lossy wireless technologies. So, the problem is to find a trade-off between application characteristics, communication technologies and protocols, cryptographic algorithms and protocols in order to satisfy security, performance, and resource consumption requirements for them.

## 1.2 Aims and objectives

The main aims of this research are to compare wireless communication technologies used for smart home applications and 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. In the second part of this research, in order to achieve the second goal we will consider a segment of a home automation system with a battery-powered sensor node, execution time and power consumption evaluation of this node.

So, in Table 1.1 you can see the main aims of this research.

| | | |
|---|---|---|
| 1 | Determine which communication technology is most suitable for data transfer in low-power, low-bandwidth home area networks. | A1 |
| 2 | Determine how security affects the power consumption of constrained IoT device and performance of the wireless communication channel. | A2 |

Table 1.1: Aims of the research

Main objectives of the thesis will accommodate:

1. Internet of Things:
   - What is the Internet of Things? Open issues in the Internet of Things.
   - What is smart home? Common applications in a smart home: Home Automation and Smart Energy. Security issues in a smart home.
2. IoT devices:
   - Parameters and constraints of IoT devices. Classification of IoT devices by constraints.
   - Examples of real world IoT devices, their parameters, and classification. Arduino and Raspberry Pi. Different Arduino microcontrollers, their parameters, and classification.
3. Communication technologies in the smart home:
   - Requirements for wireless communication technology in the smart home.
   - Performance parameters of wireless communication.
   - Comparison of most common low-power, low-bandwidth wireless communication technologies in the smart home:

– ZigBee;

– Z-Wave;

– WiFi;

– Bluetooth;

– ANT;

– Custom sub-1 GHz technologies.

- Home Automation and Smart Energy profiles in ZigBee. Security mechanisms of ZigBee.
- Wireless communication modules RF 433 MHz for Arduino, their parameters.

4. Home automation model:

- Constructing the sensor node for the home automation system based on the Arduino Nano microcontroller and RF 433 MHz wireless communication.
- Experimental measurements of execution time and power consumption of Arduino microcontroller with RF 433 MHz wireless module when transmitting data with different levels of security.
- Analysis of experimental results. Practical recommendations.

## 1.3 Research questions

As a result, it was suggested the following research questions:

Q1: What are the requirements for smart home applications?

Q2: What communication technologies are used in low-power, low-bandwidth smart home networks? Comparison of ZigBee, WiFi HaLow, Bluetooth, BLE, ANT, and Z-Wave wireless technologies.

Q3: How to build a home automation project based on cheap and easy to use Arduino microcontrollers?

Q4: How does security affect the performance and power consumption of constrained Arduino microcontroller? How to maximize the lifetime of the device?

## 1.4 Thesis scope

The scope of this thesis is limited to low-power and low-bandwidth home area networks and smart home applications with constrained IoT devices. We mostly consider low-power and short-range wireless communication technologies. In this paper we reviewed main characteristics of wireless communication technologies such as ZigBee, Z-Wave, WiFi, Bluetooth, and ANT in terms of performance, power consumption, and security. Also, we are considering the analysis of constructing the sensor node for the home automation system based on the Arduino Nano microcontroller and RF 433 MHz wireless communication in

terms of encryption and data transmission in order to continue the lifetime of the sensor node. Due to the time limitations, we only covered AES encryption algorithm as a most popular symmetric algorithm in IoT and RF 433 MHz module for Arduino.

## 1.5 Expected outcomes

The followings are the expected outcomes of the research:

1. Review of open issues in the Internet of Things.
2. Classification of smart home applications and requirements for home wireless communication.
3. Comparison of ZigBee, WiFi HaLow, Bluetooth, BLE, ANT, and Z-Wave wireless communication technologies and recommendations for their suitability for smart home applications.
4. A model of a battery-powered sensor node based on Arduino in home automation scenario and simulation of its work.
5. Evaluation of battery lifetime of the sensor node with different parameters (with encryption or without, with sleep management or without, with different communication data rate) and practical recommendations for developers of home automation projects.

It is believed that research presented in this paper will help developers and device manufacturers of smart home applications to make a deliberate choice in selecting wireless communication technology for their applications. Also, it will be useful for electronic enthusiasts who want to build their own autonomous home automation projects with battery-powered sensor nodes based on Arduino microcontrollers.

## 1.6 Thesis outline

In the next chapter, we will present a list of background and related works concerning Internet of Things and smart home concepts; communication technologies and protocols used in the IoT; home wireless communication technologies, their comparison and evaluation. In Chapter 3 we discuss open issues in the Internet of Things, parameters and classification of constrained IoT devices. Also we present the smart home, home automation and smart energy applications in a smart home. Then we consider wireless communication technologies used in smart home networks, compare them, and give recommendations. Chapter 4 shows methodology and technology used in this research and describes a model of home automation system. In Chapter 5 we discuss experiments and measurements taken and analyze obtained results. Chapter 6 gives a wrap up of the thesis and recommendations for future work.

# 2 BACKGROUND AND RELATED WORKS

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 [6] 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. Smutný in [3] attempted to consider different classifications of the Internet of Things.

Razzaque et al. [2] provide a survey about characteristics of IoT infrastructure and suggest requirements for IoT middleware which can ease the development process of IoT applications integrating heterogeneous computing and communications devices and providing interoperability.

In papers [7, 8] open issues in Internet of Things and security aspects, in particular, are considered. J. Granjal et al. [8] 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.

Giuliano et al. in article [9] present an analysis of IoT capillary networks with heterogeneous devices and propose a time-based algorithm for secure access for uni- and bidirectional devices. The proposed algorithm includes such approaches like time-based secure key generation and renewal, security for uni- and bidirectional devices, bootstrap procedure, and cognitive security concept. The article includes simulation and evaluation of offered algorithm in comparison with other technologies, and also security analysis.

In paper [10] novel efficient and collusion resistance key pre-distribution scheme based on the lightweight HIMMO scheme are presented. Performance evaluation and comparison between different security architectures are shown.

Bello et al. in [11] discuss implementation challenges in Internet of Things and solutions for network-layer interoperability between different applications. Some limitations of using TCP/IP protocol stack in IoT and limitations of 6LoWPAN architecture are present. The paper shows how to use a 6LoWPAN protocol as an adaptation layer for protocol stacks of other technologies to bring IPv6-interoperability to the IoT.

Raza et al. in [12] explored the option of using IPsec as a security mechanism for end-to-end communication in 6LoWPAN network and compared it with IEEE 802.15.4 link-layer security. An implementation and evaluation of 6LoWPAN/IPsec extension and IEEE 802.15.4 are made in the well-known Contiki operating system. An evaluation of time, power consumption, and network performance are done and compared.

Work of Bressan et al. [13] explores the implementation of smart monitoring system over the wireless sensor and actuator networks (WSANs) using RPL protocol and Web services. Based on these technologies they developed a framework for smart grid applications.

Zanella et al. provide in [14] a comprehensive survey of IoT technologies, protocols, and architecture for implementing smart city solutions in urban environments. A concept of Smart City with common applications (services) is introduced in their work. They discuss transcoding operations between constrained protocols of IoT and unconstrained protocols widely used on the Internet. This concept and architecture of the network were applied in the development of Padova Smart City.

Many researchers in their papers were trying to compare different wireless communication technologies and protocols to select most suitable for certain applications including those for a smart home.

Lee et al. in [15] reviewed security vulnerabilities of a smart home which contain resource constrained devices. Security challenges and threads are examined and security requirements for a smart home are formulated.

Andersson in paper [16] discussed "last 100 meters connectivity" of small IoT devices to services on the Internet using wireless technologies. He highlighted some parameters when selecting the appropriate short-range wireless technology for IoT. The author also compared general characteristics of some wireless networks such as applicability for some use cases, usefulness in IoT, power consumption, data rate, and distance. Some typical IoT architectures are presented in this work.

A. Rahman in paper [17] presents a broad overview and comparison study of ZigBee, Bluetooth Low Energy, Z-Wave, NFC, HomePlug GP, and WiFi short range wireless communication standards and IoT data link protocols with low power consumption, compares their main features, applications, and behaviors in terms of transmission speed, modulation type, coexistence, security, and of course power consumption.

F. Johari in [18] theoretically analyzes communication protocols currently available on the market used to implementing smart homes. Two of them, Z-Wave and ZigBee, were chosen for deep analysis which includes general information and security mechanisms on each layer, their comparison, measures taken for protection against replay and eavesdropping attacks from third parties. Then Z-Wave protocol was analyzed practically by a case study of a smart home network simulation with several Z-Wave components. Practical implementation of attacks described in the theoretical evaluation was analyzed and discussed, some vulnerability was discovered in security implementation of the protocol by the manufacturer.

C. Gomez and J. Paradells in [19] give a survey of the main current and relatively new solutions which are tailored to or most suitable for wireless home automation networks according to the authors. They considered such technologies: ZigBee, Z-Wave, INSTEON, Wavenis, and IP-based solutions that use 6LoWPAN technology with an overview of them and discussion. The main futures, use cases, and requirements for home automation networks were stated. The next applications in home automation networks are given as an example: light control, remote control, smart energy, remote care, security and safety. The detailed comparison of listed solutions is given at each layer of a protocol stack with regard to modulation and spread-spectrum techniques, available channels, link-layer reliability and delay, routing and link quality metrics, link failure detection, end-to-end acknowledgment and

retransmission mechanisms, application layer commands and attributes, security mechanisms, Internet connectivity, implementation size, standardization and market adoption, etc.

T. Mendes et al. in [20] give an updated view on wireless networking technologies for short range applications with analysis of communication requirements within smart home framework. A general framework of the proposed smart home model that integrates four major categories of smart home applications as a part of infrastructure: energy, health care, entertainment, and security is provided in the research. Wireless networking protocols supported on IEEE standards such as IEEE 802.15.1 (Bluetooth), IEEE 802.15.4 (ZigBee, 6LoWPAN, WirelessHART, MiWi, ISA100.11a), IEEE 802.11 (WiFi) and protocols that are not based on standards such as SimpliciTI, Z-Wave, EnOcean, INSTEON with their comparative assessment are presented. The suitability of listed wireless protocols to functional categories in the smart home according to their requirements is analyzed.

Neelakanta and Dighe in [21] provided methods of evaluating the performance of ZigBee and Bluetooth wireless communication when they operate in an industrial environment. They compare these two short-range wireless technologies which both operate on the same 2.4 GHz ISM band but use different modulations and analyze their robustness to interferences presented on a factory floor. They also have computed mutual interference in a scenario when these systems are operated in the same factory locale.

C. Saad et al. in article [22] present a comparison of wireless communication technologies focused on their performance and key features evaluation which depends on the limitations of communication technologies and the areas of utilization. The main goal of the paper is selecting a better communication technology that guarantees the quality of communication, minimizes energy consumption, reduces the implementation cost, etc. Authors consider parameters which influence the performance and quality of communication system in intelligent sensors and their applications. Three types of applications are considered: environment monitoring, event detection, and tracking with requirements to them. Comparative study of Bluetooth, UWB, ZigBee, ZigBee IP, WiFi, WiMax, GSM/GPRS and their characteristics using different theoretical performance evaluation metrics: network size, transmission time, transmission power and range, energy consumption, chipset power consumption, bit error rate, and data coding efficiency is presented in the article.

But despite the importance of energy-efficiency of IoT devices it is not sufficiently investigated. Furthermore, the relationship between security, performance, and power consumption needs to be studied in more detail to achieve a trade-off between them.

M. Siekkinen et al. in [23] study the energy consumption of BLE and ZigBee/802.15.4 by measuring real devices and derive a model of typical energy consumption from the measurement results. The impact of interferences on both technologies was considered. They also analyzed the scenario of using IPv6 over Bluetooth Low Energy in future 6LoWPAN deployments.

J. Suhonen in Ph.D. thesis [24] investigates QoS support in low-power wireless sensor networks with limited computation, communication, memory, and energy resources on network traffic level. The author considers maximization of network lifetime via energy-efficient protocol designs and scheduling algorithms using application specific approach to

collect and process sensor data from WSNs with high density and plenty of nodes. The paper defines a set of performance requirements for low-energy WSNs and metrics to compare different networks by QoS parameters. Also, a survey of existing QoS protocols and standards for low-energy WSNs is presented. The main objective of the research is to design adaptive communication protocols and measurement methods which will allow meeting the QoS demands of the application with some level of guarantees and the ability for the user to verify the network performance according to the stated requirements. Designed protocols were verified in simulations, prototypes were implemented in university WSN, and real-world deployment was studied.

C. Trasviña-Moreno et al. present in article [25] the design and implementation of an autonomous WiFi sensor for monitoring heating systems in the Internet of Things. Energy harvesting technology from heater thermoelectric energy was used. Viability analysis and real world tests of a WiFi sensor with energy harvesting were realized. Measurements of voltage, current, and temperature were made using different laboratory equipment and LabView for visualization. To convert measurement results to energy some calculations were made. Authors analyzed the efficiency of the system and power capabilities of energy harvesting and compared them with other similar proposals. The advantages of using WiFi instead of, for example, ZigBee, were also given.

Power saving techniques for microcontrollers are discussed in [26, 27] and tests for ATmega328 processor show the obtained effect. In [26] Arduino Uno was considered as a baseline to demonstrate running the processor at a lower frequency and at a lower voltage, turning off unneeded internal modules (for example, ADC, watchdog timer, brown-out detection, etc.), putting the processor into the sleep mode, removing inefficient linear voltage regulators and extra hardware or other energy-consuming external devices (LEDs and displays), turning off external devices when not in use (temperature sensors, SD cards), etc.

M. Levy in [28] creates a reasonable methodology for comparing different microcontrollers by their power consumption. For this purpose, a benchmark algorithm that represents a typical microcontroller application with the ability to demonstrate performance differences between microcontrollers was developed using CoreMark benchmark. The author considers 8-bit, 16-bit and 32-bit microcontrollers and compares their current consumption when running one iteration of CoreMark benchmark algorithm with values from datasheets on different working frequencies of microcontrollers and repetitive execution for different sleep time. The methodology of energy and duty cycle measurements using sleep mode is presented in the article.

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 [28] with using simple laboratory equipment like in [25] and easy-to-use Arduino electronics platform. Power saving techniques described in [26, 27] 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, 14] give the good background knowledge. For building a model of a smart home, defining its use cases and requirements

works [15, 18, 19, 20] 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. Papers [16, 17, 18, 19] give a wide overview of wireless communications used in IoT and home networks, help to understand the requirements for low-bandwidth, low-power, low-range wireless communications for smart home applications. Also, performance requirements and metrics for wireless technologies are defined in [21, 24].

# 3 THE CURRENT STATE OF THE ART

## 3.1 Open issues in the Internet of Things

There is a huge amount of heterogeneous devices such as home appliances, surveillance cameras, vehicles, sensors and actuators which work as a part of home automation, smart city, smart grid, smart agriculture, intelligent transportation system, etc. Three core components: wireless sensor networks (WSNs), machine-to-machine (M2M) communication, RF identification (RFID), and supervisory control and data acquisition (SCADA) are considered as a part of the Internet of Things [2]. It shall be able to incorporate transparently and seamlessly a large number of these different devices, link layer technologies, and services that may be involved in such a system. Now IoT goes beyond M2M communication and promises a technological revolution, but for it to work well, all the devices need to speak the same language. It's a large task which is faced with certain issues.

Let's mention some issues that various researchers are trying to solve for the development and application of Internet of Things concept in our life.

**Constrained resources**

The main issue of the Internet of Things and the main cause of other problems is that some of the devices (e.g., sensors and actuators, RFID tags) have very resource-constrained nature. They have limited computing resources, memory, and battery size. Resources of the device are mostly determined by its size, cost, and application due to the limitations in the manufacturing techniques [24]. IoT device is generally present as an embedded device, so it must have a very small size even if it's a part of a much bigger device like a car or refrigerator [29]. For some applications, IoT device must be mobile. Sometimes they even should be implanted into the body. Devices with the same size may vary in performance, but it greatly affects the price. Power usage and power availability is another main constraint. Sometimes it determines the lifetime of the device and limits the performance. Constrained device should be able to perform its function and take part in communications with other devices in the network, which can have much more resources. Constrained devices can have only simple functions and not be able to do much computation, send much data, and use heavy communication protocols and security mechanisms.

**Analysis of big data**

The second issue that arises in this scenario is where collect, process, and store data from different devices. Computations and analysis of enormous flood of data that IoT will deliver will help to convert raw data into usable knowledge [7]. Remote monitoring and control also must be provided; reactions and commands must be sent back to the devices. Cloud technology potentially could help with that or local server can be used. But in the case when we use a local database or a PC, it may have insufficient storage capacity or lack of computational resources. There will be always a risk of losing data. So, it's recommended to use more reliable cloud-based systems [20]. Although most of the IoT traffic are small and

brief like information from a sensor, it loads the network and the server, so it's better to do some filtering and preprocessing of the data at the node of the network if that node has enough resources of course. Multiple computational centers can be used at different levels of the network also [3].

**Scalability**

Experts estimate that IoT will consist of about 30 billion devices by 2020 [30]. For now, the main part of them (90% of market size [16]) are consumer devices (e.g., smartphones, laptops, TVs, and other home appliances), which are mostly concentrated in households. However, over time, there will be more devices in industrial and public sectors (discrete manufacturing, building sensor, street lights, transportation and logistics, and utilities) [11, 31].

Because of the large number of expected devices which will be soon a part of the Internet the problem of IoT devices scalability is very important. It is impossible to manage so many devices individually, so techniques of self-addressing and self-classification will be necessary [9]. Mechanism to ensure correct identification, addressing, and authentication of the devices, sensors, actuators both stationary and mobile, connected to the Internet or in a local network must be found. The first idea was to use RFID tags for identification of everything in IoT. Not only devices but also animals and people are provided with unique identifiers [6]. Domain name system (DNS) is good for identifying host and providing address mapping, it can be used in IoT although it has some security problems [7]. It is well known that the Internet will run out of IPv4 addresses quite soon, so IPv6 protocol must be used to addressing new IoT devices on the Internet. In addition to that, mobile devices can move from one network to another (can be disconnected due to poor wireless communication or battery depletion [2]), change the owner. They must be scalable, interoperable, and flexible, protected from cloning and forgery of the identifier [32]. IP-based sensor networks are emerging and will increase in the near future the capillarity of the Internet [19].

**Security**

If we are talking about security aspects, IoT devices have to be able to retrieve information and cooperate with other objects and networks without compromising security or privacy. Depending on the type of application, some security parameters of communication technology can be critical: confidentiality, integrity, availability, authentication, non-repudiation, and trust [8]. In general, communication technology in one way or another must be able to ensure these parameters even for very constrained network nodes. Some of the communication technologies have build-in security mechanisms like using Cyclic Redundancy Check (CRC), link-layer encryption, and Message Authentication Code (MAC). Despite that, cryptographic operations can be used in the application layer too. Usually, symmetric cryptography (e.g., AES) is used for data encryption and building MAC. For instance, public-key cryptography allows any pair of devices to setup a secure channel or enables accountability. However, it is computationally expensive and requires the exchange and storage of long keys, what is not always possible for constrained devices. Symmetric cryptography is faster, lightweight but it doesn't scale, so key distribution and management issue in symmetric systems, especially in IoT, is another challenge. If the network uses one symmetric key and attacker has stolen it, he can get access to every node in that network [10]. Because of the resource-constrained nature

of the IoT devices, it is always a trade-off between security and operation of the device. If devices use smaller keys than it's recommended because of the memory or processing limitation they should change them often enough. In addition, communication protocols and primitives should have protection against traditional and new for IoT threats (e.g., fragmentation attack) [33].

**Interoperability**

On the technical side, the most relevant issue consists in the incompatibility of the heterogeneous communication technologies. For example, when we send data from low-power, low-bandwidth lossy wireless networks with constrained node devices like WSN to the Internet and in the opposite direction. They may have different protocol stacks and another physical medium. Transcoding operations between the constrained and unconstrained protocol stacks can guarantee an easy way for interoperability of the IoT nodes with each other and with the Internet (HTML/XML <> EXI, HTTP/TCP <> CoAP/UDP, IPv4/IPv6 <> IPv6/6LoWPAN) [14]. EXI (Efficient XML Interchange) is a binary XML format for exchange of data on a computer network. Using EXI format reduces the verbosity of XML documents as well as the cost of parsing. Protocol CoAP (Constrained Application Protocol) is a specialized web transfer protocol for use in constrained nodes and networks. TCP (Transmission Control Protocol) is not the preferred method of communication for smart objects as it requires relatively many resources for TCP-handshake when establishing a connection [12, 15]. UDP (User Datagram Protocol) is used instead by sacrificing reliability. 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) is a protocol for using IPv6 over low power and lossy networks. It provides the capability for WSN to be identified on the Internet by a unique IP address (mesh under routing) or end-to-end IP communication between nodes (route over routing) [19]. 6LoWPAN is widely used as an adaptation layer for different technologies in IoT [8, 11, 12, 14]. Routing Protocol for Low power and lossy networks (RPL) was designed in order to match the requirements of networks characterized by low power supplies and lossy environment with high interference rate. TinyREST efficiently implements Web services on sensor networks by carefully minimizing the overhead introduced by the transport layer while using XML and WSDL data formats. So, web services can be translated into RESTful-based what is currently preferred for these networks due to its lightweight character [13]. The main idea is to make them able to communicate even if they use different physical and higher-level protocols.

**Reliability**

In data transmission, packets could fail or be damaged because of the collisions and interferences in the channel. Retransmissions and error handling can have a huge impact on the network performance. Besides, most of the communication protocols used in IoT doesn't guarantee the reliability of packet delivery [15]. Different error detection and correction algorithms have large overhead that impacts power usage and computation capabilities of constrained devices. There are various mechanisms for reducing the impact of interferences and avoid collisions. For example, frequency agile mechanisms, or using media access control protocols.

## 3.2 Constrained devices in the Internet of Things

Limitations of IoT devices determine their productivity and affect communication and security capabilities. All the required computations for running the useful application code, internal device management, communication, and security operations are performed on the same processor. Very often microcontroller and communication module share the same power source. Both communication and security compete for processor resources with functional part of the program. Both of them can be very resource intensive. IoT device must perform its immediate functions like data processing, reading and writing data to the memory, input/output operations with external devices (sensors and actuators). Device management includes running timers and counters, clock generation, resource management, event handling, etc.

So, let's consider parameters of IoT devices and their possible constraints.

**Parameters of IoT devices**

Summarizing data from different sources including RFC 7228 [34], from our point of view we can identify following core parameters of IoT devices that are shown in Table 3.1.

The main characteristics of their computation capabilities are determined by the width of the microprocessor and its speed. The frequency of the processor is counted in megahertz (MHz) or cycles per second. The width of the processor is a little more complicated parameter. There are few different parameters of a processor which can be different: data bus size, address bus size, and internal registers size [35]. According to microprocessor architecture (von-Neuman or Harvard), they can have two different data buses [36]. Often internal registers are larger than data bus, so the processor will need few cycles to fill it [35]. Address bus size can be different from registers size, but usually, the size of internal registers determines the type of processor.

According to power strategy of the device, software algorithm, and communication requirements, duty cycle determines the fraction of time during which device is active. The device can sleep most of the time to save energy and perform its functionality during a short period of time.

Communication unit connects the node to the network [24]. Communication mode defines the direction in which device need to communicate with other devices. It may be one-way communication (simplex mode) or bidirectional communication (duplex mode). Internet connectivity parameter determines if the device can directly communicate with the Internet. Data transfer rate depends on device function and determines an intensity of the communication.

User interface and accessibility show whether a user can easy configure the device in the deployment stage, set security keys, have a possibility to update software on the device, change the battery, get remote access to the device, etc [34].

| № | *Parameter* | | *Classification* | |
|---|---|---|---|---|
| 1 | Size | | Small / wearable | |
| | | | Medium / portable | |
| | | | Big / stationary | |
| 2 | CPU | CPU frequency | | |
| | | Type of processor | | |
| 3 | Power | Available power | | |
| | | Power supply mechanism | Mains-powered | |
| | | | Battery operated | Replaceable |
| | | | | Non-replaceable |
| | | | Energy harvesting | |
| | | Power strategy | Always-on | |
| | | | Low-power | |
| | | | Normally-off | |
| 4 | Duty cycle | | | |
| 5 | RAM | | C0 | |
| | | | C1 | |
| 6 | ROM/Flash | | | |
| | | | C2 | |
| 7 | Communication | Communication technology | | |
| | | Communication mode | Simplex | |
| | | | Duplex | Half duplex |
| | | | | Full duplex |
| | | Internet connectivity | IP | |
| | | | Non-IP | |
| | | Data transfer rate | | |
| 8 | Cost | | | |
| 9 | User interface and accessibility | | | |

Table 3.1: Parameters of IoT devices

**Constraints of IoT devices**

As it was noticed before, a lot of the IoT devices are small and they should be very simple, cheap and work without human intervention as long as possible. Some of them are portable or even wearable (smartwatches, heart rate, and glucose monitors) and must have limited size. As a result, small IoT devices often have the resource-constrained nature (limited processing capabilities, low memory, and limited access to power) and they are often joined to each other through a low-rate lossy wireless connection. Other entities on the network (e.g., smartphones, PCs, gateways, controlling servers) might have more computational and communication resources, so they can support interaction between the constrained devices and the Internet [34]. Devices which can directly connect to the Internet are called IP-devices, other devices are non-IP. There are six different types of IoT devices by computing capabilities: RFID/NFC tags or devices, Wireless sensor and actuator networks (WSANs),

low-end computing devices, middle-end computing devices, high-end computing devices (SCADA front-end processor), cloud. For example, passive RFID tag has no processing capacity and battery at all [2]. Flash/ROM of the device may be constrained, so it can't store large and complex programs. Because of that device may support only simple functions according to its application and light communication protocol. RAM is also limited resource. A device may not be able to store in memory large state data or use big buffers. It can limit data processing capabilities. As a result, some of the algorithms can't be run on these devices. The same applies to used communication technologies.

According to that and to the classification of constrained devices given in [34] we can say that there are three types of IoT devices:

- very constrained sensor-like devices which don't have enough resources to offer end-to-end IP connectivity at all (class C0), so they need to interact through the gateway/mediator or other IoT device which has that capability;
- quite constrained devices which can communicate with Internet but only when using specifically designed for constrained nodes protocol stack (e.g., 6LoWPAN, RPL, CoAP) (class C1);
- devices which have enough resources to support the same protocol stack (TCP, IP, HTTP, TLS) like other network devices on the Internet (class C2).

Different power supply mechanisms (mains-powered, battery operated, energy harvesting) and different strategies for power usage (e.g., using sleep mode) can be used. Devices of class C0 are usually battery operated portable devices which take care of small data from sensors. Also, all the constraints of the nodes may lead to constraints on the network [34]. For example, some nodes of the network can be unreachable for some time (depending on their duty cycle), network may have weak real performance characteristics, constraints on packet size or lack of some network functions (acknowledgement, multicast), the clear asymmetric characteristics, low penetrating ability and range if we are talking about wireless network. These devices can be both unidirectional and bidirectional. Unidirectional devices can only transmit or receive data in one direction (sensors and actuators), so there will not be acknowledgeable communication between the nodes. Some of them can't be properly coordinated and receive any commands, so they can be unsynchronized and interfere with other communications in the network [9]. Their configuration and management may be difficult and may require physical access to the device.

RFC 7228 gives next values of parameters RAM and ROM shown in Table 3.2 for their classification of constrained devices. IoT devices may be also classified by their main purpose to high performance and low energy devices [24].

| *Number* | *Description* | *Class* | *RAM* | *ROM/Flash* |
|---|---|---|---|---|
| 1 | Non-IP | C0 | « 10 Kb | « 100 Kb |
| 2 | IP, constrained protocol stack | C1 | 10 Kb | 100 Kb |
| 3 | IP, usual TCP/IP stack | C2 | 50 Kb | 250 Kb |

Table 3.2: Classification of IoT devices by constraints

C. Lee et al. in [15] give a list of common devices for the Internet of Things and particularly for the smart home. Some of them are shown in Table 3.3.

| *Device Type* | *Chipset* | *CPU frequency* | *RAM* | *ROM/Flash* | *Power supply mechanism* | *Communication technologies* |
|---|---|---|---|---|---|---|
| iPhone | A7x Quad-core Processor | 1.7 GHz | 2 Gb | up to 128 Gb | battery | WiFi, Bluetooth, NFC |
| Samsung Smart TV | Exonys SoC | 1.3 GHz | 1Gb | - | mains-powered | WiFi |
| Nest Learning Thermostat | ARM Cortex-A8 | 800 MHz | 512 Mb | 2 Gb | battery | WiFi |
| Nest Smoke Detector | ARM Cortex-M0 | 48 MHz | 16 Kb | 128 Kb | battery | WiFi |
| Fitbit Smart Wrist Band | ARM Cortex-M3 | 32MHz | 16 Kb | 128 Kb | battery | Bluetooth LE |
| Philips Hue Light bulb | TI CC2530 SoC | 32MHz | 8 Kb | up to 256 Kb | battery | ZigBee |
| Sensor Devices | Microcontroller | 4-32 MHz | 4-16 Kb | 16-128 Kb | battery | ZigBee, WiFi, Bluetooth |

Table 3.3: Typical IoT devices

As you can see from the table some of the devices such as sensors and detectors, wearables, and light bulbs have restricted CPU and memory. There are devices of C1 class and C0. But almost all devices have limited power supply, even iPhone. For communication ZigBee, WiFi, and Bluetooth are mostly used. Sensor devices most likely are developed on simple microcontrollers like Arduino.

## 3.3 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. Smart home vision aims for the integration of all these devices, services, communication and information technologies in one system allowing users to control them from any room in the house or any location in the world using Internet connection [20]. The home technology is developing rapidly to integrate all home systems under one centralized management system which forms Home Area Network (HAN) [37]. 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. Classification of main systems of smart home based on [3, 19, 20, 38] are shown in Figure 3.1. 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. Some entertainment devices may be used like a convenient tool for management of home automation system or smart energy system. From IoT perspective, it's also possible to work from any device, anywhere, and anytime. It can be a smartphone, a smart TV, or a remote control. Wearables may be used for monitoring babies, sick or elderly people, as well as for sport and fitness. HAN has access to the Internet using home gateway and can communicate with other services in or outside the cloud like smart city, smart grid, and smart healthcare [20, 38]. The smart grid can also have alternative ways for communication with smart homes [38, 39].

Smart home can include a lot of different systems and applications, as illustrated in Figure 3.2.

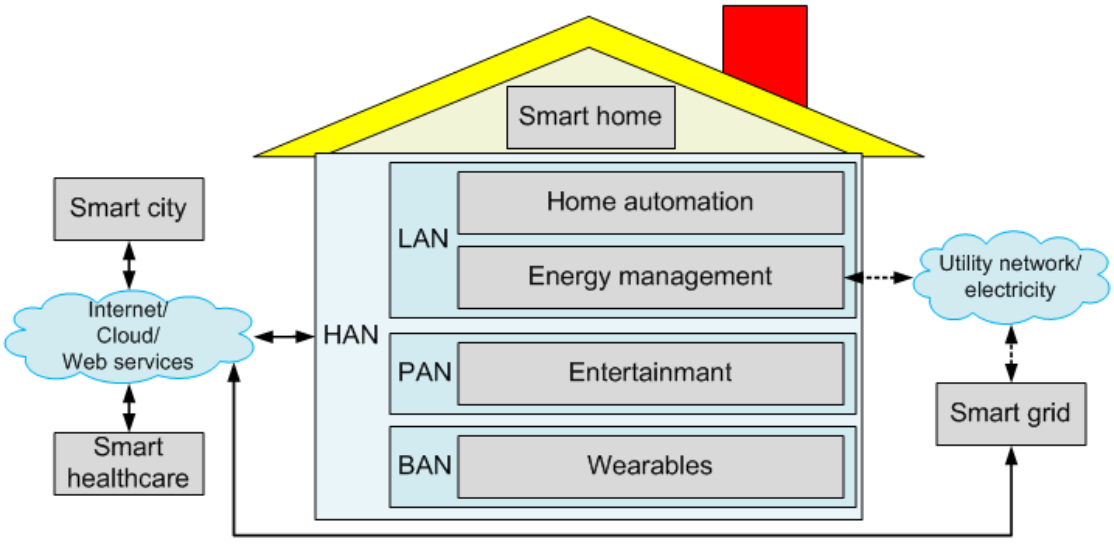



Figure 3.1: Structure of smart home

**Home automation system**

Home automation is the control system of any or all electrical devices inside our home to improve comfort, security, and save energy. We can consider home automation like an extension of building automation. And it's also a part of a smart home. It should be able not only control and monitor home devices remotely, but also be more flexible and intelligent, learn preferences of residents and anticipate their desires.

A classical example of home automation application is smart lighting system [40]. Lighting is a major expenditure for electricity, therefore the savings are needed. Home automation system may adjust light depending on the amount of daylight in the room and availability of other light sources, type of the room, presence of people in the room, individual settings of brightness and color for each person. In such a way we can significantly reduce lighting costs. The light control system may include smart bulbs, paired with lighting switches, light sensors (photocells), remote controls, occupancy and vacancy sensors, etc. Light sensors and switches may be powered by energy harvesting technology, for example, a light sensor may use solar batteries and a switch may harvest energy from flipping the switch on or off. It may be used to control light from your smartphone, you can even turn it on periodically being on vacation to protect the house from burglars.

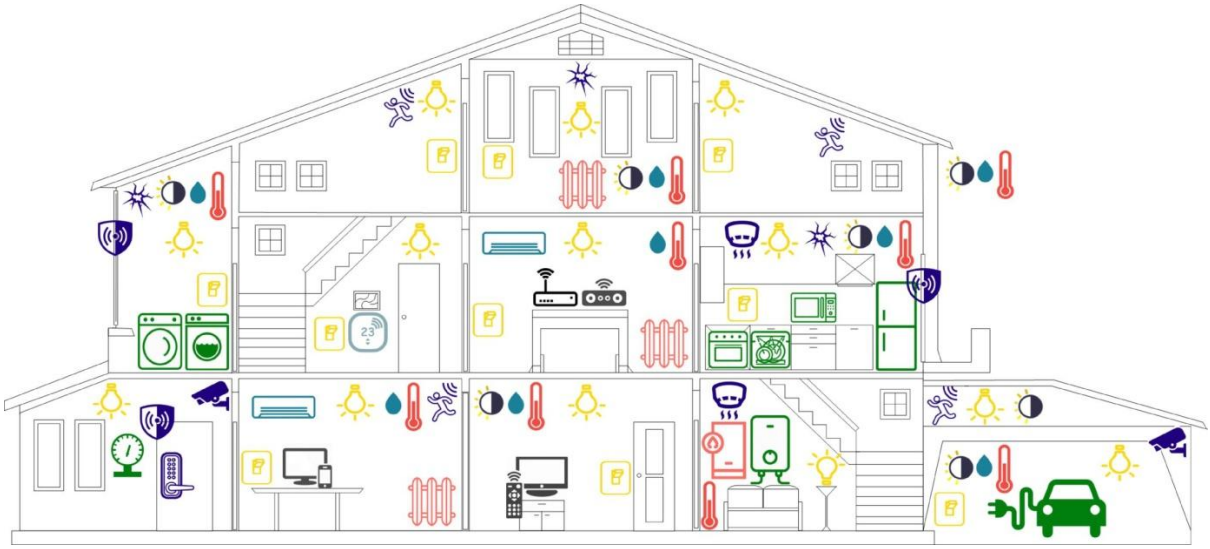


Figure 3.2: An example of smart home

HVAC system may include temperature and humidity sensors all over the house and even outside, thermostats, smart air conditioners, remote controls, heaters, radiators, etc. This system will allow to adjust the microclimate in the room (by regulating the speed, temperature, operation modes of air conditioner, fan direction, temperature of the heater, regulation of radiators in each room, etc.) according to the current microclimate parameters, weather outside, season, presence of people in the room and their preferences. Temperature sensors can also use energy harvesting techniques like thermoelectric energy from the heater to measure its current temperature, solar energy for sensors outside the house. You can turn off the heater from the smartphone when leaving the house and turn it on again before coming back from work, or set schedule. During the vacation, you can warm up your house periodically, especially in the winter.

Another important part of home automation system is home security. It includes security and door cameras, door and gate locks, motion detectors, opening and glass break detectors, smoke and water leak detectors, etc. You can check video from IP-camera using PC or TV inside your home or remotely, unlock your door with a smartphone. In a case of emergency, you can receive a message to your phone or alarm may be activated.

There are many other examples of using Internet of Things devices in home automation. For example, automatic doors and curtains.

**Energy management system**

The idea is quite simple: some smart appliances which usually consume much power can operate when electricity is the cheapest (during the night or off-peak time) [41]. This system works very closely with utilities and governments. For practical implementation of such a system, smart grid infrastructure must be built, utilities must constantly broadcast pricing information to users. Smart meter inside user's home can help to monitor and control energy information coming to and from your home, for example using power line communication (PLC), by home energy management system [42]. Utility gets information how much electricity is being used. Pricing information from the utility is spread between home

appliances through the home network, for example using ZigBee, and they try to adjust their working time (for example, washing machine, dishwasher or microwave oven can go into delayed start mode or economy mode, washing machine and dryer can share laundry settings). You can monitor your current energy usage by each appliance separately and change a schedule and savings strategy. It's also possible to use your own small-scale home renewable energy systems (e.g., solar panels and wind turbines) and energy storages (e.g., plug-in electrical car) and even sell excess of the energy back to the utility [42].

The problem is when your door lock or security camera uses the same level of security like a bulb or a sensor. But losing control over the temperature sensor can also be dangerous. In that situation, an attacker can affect the behavior of heating and cooling devices, turn them on and off anytime, run them in extreme modes, and, as a result, cause huge electricity bill and even physical damage. If the attacker deactivates the door lock he can get physical access to the house.

## 3.4 Communication technologies in the Internet of Things and smart home

All the challenges mentioned in Subsection 3.1 rely on efficient data communication between heterogeneous IoT devices (i.e., bidirectional, unidirectional, IP and non-IP) [9]. Communication technologies have their own limitations, which can be combined with constraints of the nodes in the network. They can be wired or wireless, have different range (max distance), data rates (speed), work on different frequencies, use various transmission medium, have different energy consumption requirements, have diverse resistance to obstacles and interference, different characteristics, they have different standards and use different protocol stacks which have different embedded security mechanisms and security issues. Some of them may have obvious advantages or disadvantages in comparison with other for using them for a specific purpose (short distance or long distance, large throughput, low packages loss, tolerable delay, acknowledgment and reliability, low energy consumption or high transmission power, a cost of the equipment, strong embedded security, low protocol overhead). The problem is how to compare them. A wide variety of communication technologies are used in the Internet of Things word: ZigBee, Bluetooth (BLE), WiFi (WiFi-direct, WiFi HaLow), ANT, Z-Wave, DECT ULE, INSTEON, Wavenis, Thread, LiFi, Ethernet, Cellular (2G/3G/4G), DASH 7, Weightless, UWB, WiMax, Sigfox, Neul, LoRaWAN, PLC, IrDA, NFC, RFID, QR codes and barcodes [11, 16, 43, 44], which are used at different levels of Internet of Things networks: Personal Area Network (PAN), Local Area Network (LAN), Metropolitan Area Network (MAN), and Wide Area Network (WAN).

Many IoT devices are using wireless connectivity for M2M communication since it offers flexibility, mobility, scalability and cost efficiency needed for sensors and actuators networks. There is no need to lay cables with conduits or cable trays [19]. Radio frequency (RF) communication seems to be a good idea. It has developed infrastructure and includes a variety of technologies for different application needs, doesn't require line of sight [24, 45]. But wireless communication has also some disadvantages with respect to the cable network.

Technological challenges involved in wireless networks is not trivial. As an example, lower reliability and data rates due to interference, security threats since the signal is spreading through the air and anyone within range of the network can easily intercept it, higher power consumption, etc [46]. But for our smart home scenario listed disadvantages are not essential because we can accept lower performance, minimize power consumption of our RF devices, or improve protection of our communication.

### 3.4.1 Requirements for home wireless communication technologies

When a consumer/manufacturer advisedly chooses wireless communication technology to interconnect his IoT devices he may consider next requirements:

- *low cost* (cost of the radio must not be much higher than a price of the IoT device, better when no additional infrastructure is required);
- *low power consumption* (usually radio must share the same power supply with sensor and microcontroller, network lifetime should be maximized);
- *ease-of-use* (simple architecture, it must be easy to configure the network and add a new device to it, nodes are autonomous and self-configurable, minimal human intervention required, possibility to control and monitor IoT devices remotely using Internet);
- *security* (embedded security mechanisms for confidentiality, integrity, authentication with a possibility to choose the level of security required for application);
- *sufficient range* (wireless communication network must cover all IoT devices including portable inside or around the house, different network topologies may be used, peer-to-peer communication and multihop communication must be allowed, better when no additional infrastructure (e.g., repeaters) is required, better when fewer neighboring houses are affected by your home network);
- *interoperability* (widely used standardized technology, different devices from different manufacturers may be connected);
- *scalability* (network can have hundreds of nodes with high density, new devices may be added, the size of the network may be increased);
- *sufficient network performance* (throughput, packet loss, latency) for all nodes, resistance to obstacles and interferences.

Mats Andersson in his work [16] suggests first six requirements but doesn't consider quality parameters (QoS). QoS is usually understood as a set of traffic performance requirements for data flow in communication networks. Performance requirements are usually expressed with throughput, delay, and error rate metrics [24]. From the other side, authors of [45] consider next requirements: throughput, power consumption, and range. It is noteworthy that home wireless network needs to support the varied quality-of-service requirements for different applications. In fact, many applications require real-time performance (e.g., healthcare applications, online games, home entertainment systems, PC peripherals) or max speed (e.g.,

audio and video streaming). The realization of QoS requirements is imposed on communication protocols, which may have different level of guarantees. There are two main classes of communication protocols: contention and contention-free protocols. While contention-based protocols allow flexibility, contention-free mechanisms provide energy-efficiency and reliability [24]. So, their combination may be used for adjusting QoS parameters. Routing protocol has also a significant impact on QoS.

From the practical point of view, it's quite difficult to determine the real network characteristics like range and throughput, because there are too many uncertain parameters for different network configurations, environment parameters, and obstacles. The manufacturers of wireless communication devices usually include in the datasheet of their devices information about range and speed (data rate), which are either theoretical maximum or measured in conditions close to ideal (outdoor in line-of-sight). We can not really tell which method did the manufacturer used. These parameters may only be the start point for the rough comparison. So, it's recommended to test your network in place after configuration to determine whether the network meets the minimum requirements of the application.

**Physical aspects of wave propagation**

To understand better what affects real characteristics of the network and the wireless technology we need to know more about radio waves propagation. This will also help design your network and improve its characteristics. Electromagnetic waves propagate through environment from a transmitter to a receiver with speed of light. To transfer data between them we manipulate the radio waves to modulate our signal. Different types of modulation can be used, such as ASK, FSK, PSK, and more complex. While wave is propagating through space it's changing because of scattering, reflection, refraction, absorption, diffraction, and attenuation [47, 48] caused by walls, building, trees and other obstacles. This effect is called shadowing. Reflected, diffracted, or scattered copies of the transmitted signal can have slightly different parameters from the original signal. They can be attenuated in power, shifted in time, have changed frequency and phase. These signals (multipath components and original signal) combine together at the receiver side and we get distorted signal. This effect is called multipath. Furthermore, the main effect called path loss is power dissipation when a signal is propagating over the distance [49]. Transmission energy is proportional to the square of the distance [24]. To compare the quality of the signal usually next three parameters are used: signal strength, noise floor and signal-to-noise ratio (SNR) [47].

Indoor environments like houses differ widely in materials used for walls and floors, an arrangement of rooms and their sizes, a location of windows and doors, number of floors, materials and location of furniture and other obstacles [49]. In a case of a home automation system, smart devices can be scattered throughout the house and all these factors will have an impact on quality of the communication between them.

The main characteristic of a radio wave is its frequency or wavelength. Radio waves have frequencies as high as 300 GHz to as low as 3 kHz. A frequency determines the behavior of a wave. The higher frequency has shorter range and lower penetration ability but we can transmit more data over a high-frequency wave and get more performance. Different wireless technologies use different frequencies and various modulation schemes (physical layer). On

top of that, they may use different upper-layer protocols (link layer, media access control layer, network layer, application layer) for framing, error detection and correction mechanisms, neighbor discovery, routing, encryption, data transferring which improve network characteristics (reliability, scalability, security) but increase communication overhead. Most of the frequencies used for wireless communications are from unlicensed industrial, scientific, and medical (ISM) radio bands. 2.4 GHz band is especially popular. But the problem is that many different devices like cordless phones, baby monitors, microwaves, car alarms, video devices and many different technologies like Bluetooth, ZigBee, WiFi use the same 2.4 GHz frequency causing interferences and as a result decrease in performance. So, it's better to use relatively interference-free bands.

**QoS parameters**

*Latency* or *delay* indicates the time elapsed since the packet generation at one node (source) to its reception at the other node (destination) [24]. The delay depends on transmitting speed at the source node, signal propagation speed through the channel, and package processing speed at the destination node, the size of the packet, the distance between the receiver and the transmitter and count of hops between them. Latency is controlled by priority-based channel access and very affected by scheduling and duty cycling of end nodes [24].

*Throughput* indicates the amount of application payload successfully delivered from a source node to a destination node over a communication channel per time unit. The throughput can be significantly less than the data rate because of the packets loss and protocol overhead.

*Reliability* denotes the probability that a packet is successfully delivered from the source to the destination. *Packet loss* is a percentage of packets lost with respect to packets sent from the source node. It is an opposite characteristic of the communication channel to reliability. Packet loss affects throughput and if reliable communication is needed packet loss causes the increasing of latency due to retransmissions. Reliability is mainly ensured by controlling a number of retransmissions [24].

The *lifetime* of IoT device determines the time from deployment to its breakage or depletion of the power source. It is an opposite characteristic to *power consumption*. The lifetime has a significant impact on other QoS parameters and there is a trade-off between power consumption and network performance.

*Security* is the level of resistance to threats, protection from intrusions and interference whether accidental or malicious that can compromise privacy and safety. In IoT security is inseparable from safety.

There are a lot of security threats inherent in wireless technologies and smart home devices. We can divide them into device security and network security. As an example, tampering, code injection, information extraction are device level threats and sniffing, jamming, replay attack, back-off attack, ACK attack, back hole attack, etc., are network level threats [15]. Some of the attacks aim to deplete battery of devices by forcing them continuously retransmit data, some of them aim to prevent any communication in the network, get access to security material inside the device and information in the network. Security mechanisms in IoT must ensure data confidentiality, integrity, and availability, provide services like key generation

and distribution, device and user authentication, physical protection, and monitoring throughout all the device lifecycle including device booting and initialization, operation, maintenance, and decommissioning. Security level should be adequate to existing threats. It may have a negative impact on other QoS parameters.

### 3.4.2 Review of home wireless communication technologies

Let's consider popular wireless technologies in the Internet of Things which we can often meet in different IoT devices in our homes. We also will consider some promising modifications of these technologies designed to expand their possibilities and make them closer to the Internet of Things.

We can classify them by range, data rate, and power consumption [16]. The possible network configuration like network topology and physical size constraints is also important for designing a network. As a matter of fact, wireless connectivity is not dominated by one single technology. In most cases, the technologies which provide low-power, low-bandwidth communication over short distances, operate on unlicensed spectrum, have limited quality-of-service (QoS) and security requirements will be widely used for home and indoor environments [45]. Following low-power, low-bandwidth technologies are most suited for this description: ZigBee, WiFi HaLow, Bluetooth, BLE, ANT, and Z-Wave. In Table 3.4 you can see the main characteristics of considered technologies. Each technology has advantages and limitations. The most common for smart home applications which are often mentioned in the literature are next technologies: ZigBee, WiFi, Bluetooth, and Z-Wave. We will examine these technologies in detail in order to determine which one is best suited in HAN for a particular application. And in Subsection 3.4.2.2 we will concentrate on ZigBee, because it is one of the leading players in a smart home market, providing low-power, low-bandwidth mesh connectivity for both home automation and energy management applications.

| | | ***ZigBee*** | ***WiFi HaLow*** | ***Bluetooth*** | ***BLE*** | ***ANT*** | ***Z-Wave*** |
|---|---|---|---|---|---|---|---|
| *Standardization* | | IEEE 802.15.4 | IEEE 802.11ah | IEEE 802.15.1 | IEEE 802.15.1 | Proprietary | Proprietary |
| *Frequency* | | 2.4 GHz, 868, 915 MHz | 900 MHz | 2.4 GHz | 2.4 GHz | 2.4 GHz | 900 MHz |
| *Range, m* | *indoor* | 10-100 | < 700 | 1, 10, 100 | 50 | < 30 | 30 |
| | *outdoor* | | < 1000 | | | | |
| *Data rate* | | 20, 40, 250 Kb/s | 150-400, 650-780 Kb/s | 1, 2, 3 Mb/s | 1 Mb/s | 1 Mb/s | 9.6, 40, 100 Kb/s |
| *Throughput* | | 10-115.2 Kb/s | > 100 Kb/s | 0.7-2.1 Mb/s | 305 Kb/s | 20 Kb/s | - |
| *Power consumption, mA* | | < 40 | - | < 30 | < 12.5 | < 16 | < 23 |
| *Tx output power, dBm* | *from* | -3 | 10 | -6 | < 19 | -20 | < 0 |
| | *to* | 10 | 30 | 20 | | 0 | |
| *Multiplexing* | | DSSS | OFDM | FHSS | FHSS | TDMA | FHSS |

| | | | | | | |
|---|---|---|---|---|---|---|
| *Modulation* | OQPSK, BPSK | BPSK, QPSK, 16-QAM, 64-QAM, 256-QAM | GFSK, $\pi/4$-DQPSK, 8DPSK | GFSK | GFSK | FSK, GFSK |
| *Security algorithm* | AES-128 | WEP, WPA, WPA2 | E0, E1, E21, E22, E3, 56-128 bit | AES-128 | AES-128, 64 bit | AES-128 |
| *Topology* | star, tree, mesh | one-hop | p2p, scatternet | p2p, star | p2p, star, tree, mesh | star, mesh |

Table 3.4: Comparison of wireless communication technologies

From the Table 3.4 you can see that BLE, ANT, and Z-Wave wireless technologies consume the least electricity, WiFi HaLow has the largest range, and Bluetooth allows the best throughput (if we don't take WiFi into consideration). Also, you can mention that security of almost all technologies is based on AES-128 encryption.

#### 3.4.2.1 Detailed comparison

**ZigBee**

ZigBee is a standard for low-power, low-rate wireless communication which aims at inter-operability and encompasses devices from different manufacturers. Protocol stack provided by ZigBee is open source and free to use by any developer or company. "Zigbee is the only global, standards-based wireless solution that can conveniently and affordably control the widest range of devices to improve comfort, security, and convenience for consumers" [50].

ZigBee is built upon the physical layer and medium access control layer defined in the IEEE 802.15.4 standard. It uses three unlicensed frequency bands depending on location: from 2400 MHz to 2483.5 MHz, from 902 MHz to 928 MHz, and from 868 MHz to 868.6 MHz. 16 channels are allocated in 2.4 GHz band, 5 MHz each, 10 channels – in 915 MHz band, and only one channel in 868 MHz band. In the 868 and 915 MHz bands binary phase-shift keying (BPSK) is used, in 2.4 GHz – offset quadrature phase-shift keying (OQPSK). A direct sequence spread spectrum (DSSS) modulation helps to increase robustness against even strong interferences [51]. A typical range of ZigBee is around 10-100 m, depending on power and environment characteristics. ZigBee has three stack profiles: ZigBee, ZigBee PRO, and ZigBee IP. ZigBee PRO comparing to ZigBee is optimized for larger networks, but it is more expensive and requires more memory. ZigBee Pro can easily support networks of thousands of devices while ZigBee supports networks of hundreds of devices. Zigbee profile doesn't support fragmentation and frequency agility, but it's still compatible with ZigBee Pro. Devices from one network can participate in another network [52]. ZigBee IP is a new technology that doesn't allow backward compatibility. It provides IP-connectivity for ZigBee devices.

More about ZigBee in 3.4.2.2.

### Z-Wave

Z-Wave is the world market leader in wireless control with over 70 million products sold worldwide, supported by over 450 manufacturers [53]. The main purpose of Z-Wave is to allow reliable transmissions of short messages from a control unit to other nodes in the network [19]. The Z-Wave protocol is an interoperable, wireless communication technology designed for control and monitoring applications for residential and commercial environments [53]. Z-Wave network consists of one gateway, one controller, and at least one controlled device. There are two types of devices: controllers and slaves. Controllers send commands to the slaves which reply to the requests or execute the commands. Slaves may act like routers in time-critical and event-based applications like alarms [19]. The Z-Wave network allows full mesh topology without the need for a coordinator [53]. The maximum size of the network is restricted to 232 nodes. Physical and media access control layers are defined in ITU-T Recommendation G.9959. The maximum range is 30 meters. Depending on region Z-Wave use slightly different frequencies: 868.42 MHz for Europe, 869 MHz for Russia, 908.4 MHz for the USA, 919.8 for Australia and Brazil, etc. The protocol was developed by ZenSys, licensed by Sigma Designs, and is promoted by the Z-Wave Alliance. All commercial products using Z-Wave technology must be certified. Since it operates in the sub-1 GHz band it is not sensitive to interference from WiFi and other wireless technologies in 2.4 GHz range [53]. Z-Wave uses a very robust frequency key modulation (Gaussian frequency-shift keying). For Europe, Manchester encoding is used, for the United States – Non-Return-to-Zero (NRZ). The maximum allowed payload size of the packet on physical layer is 64 bytes [18]. It allows transmission at data rates 9.6 Kb/s, 40 Kb/s, and 100 Kb/s using frequency shift keying modulation. It recently also supports IPv6 and uses AES-128 encryption [19].

### WiFi

WiFi technology is designed for connecting electronic devices in a wireless local area network (WLAN). WiFi is based on the IEEE 802.11 family of standards which operate in the 2.4GHz and 5 GHz unlicensed bands available worldwide. Standards IEEE 802.11b/g/n use 2.GHz band, while IEEE 802.11a/n/ac work at 5GHz. Using 14 partially overlapping 22 MHz wide bands in 2.4 GHz frequency, WiFi has a massive bandwidth, and, as a result, allows to achieve very fast data rates. Only 13 of these channels are available in Europe, 11 in the USA, and just one in Japan. The data rate is 54 Mb/s or even more [54]. Direct sequence spread spectrum (DSSS), Complementary code keying (CCK), and orthogonal frequency-division multiplexing (OFDM) are used in WiFi. WiFi uses carrier sense multiple access with collision avoidance (CSMA/CA) channel access protocol, and, optionally, a request to send/clear to send (RTS/CTS) mechanism. The WiFi Alliance is the non-profit organization managing the technology and certifying WiFi devices.

There are more than 7 billion devices with WiFi technology in use today. In the Internet of Things world, WiFi is used for remote wireless monitoring and management of lights, power outlets, door and garage locks, surveillance, presence detection, alarms, appliances, climate

control, metering, manufacturing control and diagnostics, medical and fitness equipment, and M2M data flow [54].

Since WiFi technology dominates in consumer electronics segment, it's always better to reuse the existing WiFi infrastructure for new IoT applications, especially if they need to communicate with smartphones, laptops, tablets, TVs, game consoles, etc. In addition, WiFi has a native compatibility with IP. That means that there is no need to buy expensive gateways to connect your IoT devices to the Internet. The cost savings are significant, that is why it's expected that future electronics market will boost WiFi's presence in IoT [55]. The main disadvantage of WiFi comparing to its competitors in the smart home scenario is relatively higher power consumption. To compete with ZigBee, BLE, and Z-Wave some low-power WiFi modules for embedded systems with energy-efficient radio transceivers have recently been developed [55]. That technology is called *Low-power WiFi* or *WiFi HaLow*. From the other side, WiFi is not capable of meshing, that's why connectivity is limited by access point coverage. *WiFi Direct* or *WiFi P2P* was recently developed to change that.

*WiFi HaLow* is the designation for IEEE 802.11ah technology. WiFi Alliance announced WiFi HaLow in January 2016. It operates in 900 MHz frequency band, offering longer range and lower power connectivity for WiFi devices [56]. Because of this, WiFi HaLow can be used in IoT applications like sensors and wearables. It also has some more features: the ability to penetrate walls and other physical barriers more easily, and sleep mode which helps to extend battery life. But current WiFi routers and smart devices is not compatible with WiFi HaLow, you will need a new hardware.

*WiFi Direct* is a technology that allows WiFi devices to connect directly without requiring a wireless access point. That technology allows transferring data between two or more devices without joining the network. WiFi Direct can provide a wireless connection for peripherals like wireless mice, keyboards, displays, and remote controls. It can be used to share files, print documents, or wirelessly communicate between devices quickly and easily at typical WiFi speed.

*WiFi Aware* is a standard for real-time and energy-efficient service discovery mechanism which runs over WiFi. It doesn't require GPS or Internet connection. It was developed to run background and continuously scan the environment for information and services within WiFi range and share small pieces of information in proximity [56].

The base security profile for WiFi is Wireless Equivalent Privacy (WEP) protocol, which is based on RC4 stream cipher. WEP protocol was discovered to be insecure. Instead, WiFi Alliance defined Wireless Protected Access (WPA) [46]. Next version known like WPA2 uses AES instead of RC4 and a strong authentication protocol EAP-TLS.

**Bluetooth**

Bluetooth technology is based on the IEEE 802.15.1 standard for short-range wireless communication between fixed and mobile devices in PANs based on low-cost transceiver microchips in each device. It also works in 2.4 GHz band sharing an overcrowded spectrum with other technologies. The data rate is 3 Mb/s, and 24 Mb/s are supported in v3.0 over a

collocated link. A frequency band of Bluetooth is from 2402 MHz to 2480 MHz or from 2400 to 2483.5 MHz with 79 channels, 1 MHz per channel. But only 23 channels are allocated in France, Spain, and Japan. Bluetooth radio transceivers hop from one channel to another using frequency hopping spread spectrum (FHSS). It also avoids busy channels.

Bluetooth has a master-slave structure. A collection of Bluetooth devices in one channel forms a piconet. Bluetooth supports up to 8 devices in a piconet, one of them is a master. Master determines pseudo-random function and sets the clock. Linking multiple collocated piconets using common nodes allows creating a scatternet. Any two nodes in a scatternet can communicate to each other even if there is no direct connection. Nodes on the path between them will relay messages. Each piconet inside a scatternet has its own pseudo-random function for minimizing interferences between them. Master starts its transmission in even slots, a slave in the odd ones. Packets may be 1, 3, or 5 slots long. Bluetooth uses Gaussian frequency-shift keying (GFSK) modulation, but also differential quadrature phase-shift keying ($\pi/4$-DQPSK) and differential phase-shift keying (8DPSK) modulations may be used to increase communication data rate.
Classic Bluetooth is used for streaming data based on dedicated application profiles like the audio distribution profile (A2DP) and the headset profile (HSP), for file exchange between devices file transfer profile (FTP) is used [54]. Bluetooth specification is quite complex and supports more than two dozen voice and data services [57]. In order to be able to connect two Bluetooth devices, both must support the same Bluetooth profile. Bluetooth Special Interest Group (SIG) manages the technology. There are over 30,000 members of SIG today. Primary use cases: mobile phones, PCs, printers, headsets, joysticks, mice, keyboards, stereo audio, automotive. Many Bluetooth applications are quite simple, easy to setup, and focus on the consumer.

*Bluetooth Low Energy* (*BLE*) also known as *Bluetooth Smart* was introduced in Bluetooth v4.0 specification. It is developed for battery-operated devices [54]. It uses 39 channels instead of 79, channel width is 2 MHz instead of 1 MHz, has lower power consumption and lower data rate, and doesn't provide backward compatibility. Scatternet is not supported also, only point-to-point and star topologies. However, unlike classic Bluetooth, it can support an unlimited number of nodes. Mostly used for mobile phones, PCs, gaming, sport and fitness, industrial, medical, home, automation electronics, wearable devices, and proximity tags [52]. All major mobile and desktop operating systems now support Bluetooth Smart Ready. The market of Bluetooth-enabled IoT devices is growing rapidly.

*Bluetooth 5* was introduced by Bluetooth SIG in June 2016. It is focused on increasing the functionality of Bluetooth for IoT. Bluetooth 5 has quadruple the range, double the speed, and increase the data broadcasting capacity. They also improved interoperability and coexistence with other wireless technologies.

A process of establishing a connection between Bluetooth devices is called pairing. Pairing often involves some level of user interaction (entering a PIN code, numeric comparison, entering a password, etc.). The Bluetooth security depends largely on the length and randomness of the password or the PIN code used during pairing of devices [57]. The length of the PIN code may be between 1 and 16 octets. In some cases that value may be hard coded

into the device. There are three security models: non-secure; service level security; link level security [46]. Some services may require authentication or authorization. To ensure confidentiality of the data Bluetooth uses E0 stream cipher for encrypting packets. For key generation during the pairing process E22 algorithm is used and it depends on pairing PIN. Other security algorithms based on SAFER+ block cipher are also used. It's recommended to make devices visible only when needed and remove paired devices and their link keys when not in use.

**ANT**

ANT is a proprietary protocol for monitoring and control applications with low power consumption. It is oriented to work with sensors to form multicast WSNs. ANT operates in 2.4 GHz band. It uses virtual channels and works on frequency band from 2400 MHz to 2524 MHz with 124 physical channels with a width of 1 MHz each. ANT packet has 8-byte payload and it is transmitted in 150 microseconds or less [58]. As a result, technology supports data rates of 1 Mb/s. Every network has a unique identifier to distinguish different networks. Multiple virtual channels can coexist on a single frequency. Each ANT node connects to other nodes through dedicated channels. There are three different types of channels: independent, shared, and scan channels [58]. Each node can participate in up to 8 channels. Some channels can be shared between multiple nodes. Every channel is bidirectional, it must have a master who transmits data and at least one slave who receives. In order to communicate master and slaves should determine channel identifier, agree on frequency, channel period and transmitter power [52, 58]. Different channels can have different directions. Channels adapt their transmission timeslots automatically. Frequency agility and self-adjusting isochronous TDMA technology are used in order to avoid collisions in the channel. The protocol defines physical layer, data link layer, and transport layer. Next layers are defined by vendors in extensions known as ANT+ profiles. ANT+ allows interoperability of ANT devices. These profiles determine session layer, presentation layer, and application layer according to the specific use cases [58]. ANT supports various network topologies like broadcast, peer-to-peer, star, shared, shared cluster and practical mesh with data acknowledgment and burst options [52]. It is used for sports and fitness applications like heart rate monitors, watches, distance and speed monitors, for medical, home automation, or industrial applications.

**Custom sub-1 GHz technologies**

ISM unlicensed frequency bands below 1 GHz are widely used mainly in industrial, home and building automation and metering applications. These frequency bands offer flexibility in selecting physical layer characteristics and a possibility to develop a proprietary protocol for the dedicated use case, achieving high protocol efficiency. There is no limitations in network topology or number of connected nodes. Sub-1 GHz RF allows achieving unmatchable performance and efficiency sacrificing interoperability [54]. As an example of using sub-1 GHz systems, we can mention Wireless M-Bus. It is an open standard developed for gas, water, and heat metering applications. This standard is using for the networking and remote reading of utility meters in Europe. Smart grid devices require robust, long range wireless communication. 868 MHz, 434 MHz, and 169 MHz are the most commonly used frequencies.

Using one of these frequencies allows wireless radio waves to reach difficult areas like underground meters [39]. Sub-1 GHz RF can also be used for smart city applications like light, parking and traffic systems [54]. There are a lot of low-power sub-1 GHz modules with a wide range of supported frequencies, modulation schemes and data rates from 1 to 500 Kb/s [54].

| *Smart home applications* | | *Requirements* | | | | | | | *Recommended wireless technology* |
|---|---|---|---|---|---|---|---|---|---|
| | | *Low power* | *Low cost* | *Security* | *Range* | *Topology* | *Network density* | *Throughput* | |
| *Home automa-tion* | *Lighting* | +- | + | +- | PAN/LAN | p2p, star, mesh | + | Low | ZigBee, BLE, Bluetooth, Z-Wave, ANT, WiFi HaLow, WiFi |
| | *HVAC* | +- | +- | +- | PAN/LAN | p2p, star, mesh | +- | Low | ZigBee, BLE, Bluetooth, Z-Wave, ANT, WiFi HaLow, WiFi |
| | *Security* | +- | + | + | PAN/LAN | p2p, star, mesh | + | Low, upper medium | - ZigBee, BLE, Bluetooth, Z-Wave, WiFi HaLow, ANT, WiFi (low)<br>- Bluetooth, WiFi (upper medium) |
| *Energy management* | | - | +- | + | LAN | p2p, star | +- | Low | ZigBee, WiFi, WiFi HaLow, Bluetooth, Z-Wave, BLE, ANT |
| *Entertainment* | | +- | +- | +- | PAN/LAN | p2p, star | +- | Upper medium, high | - Bluetooth, WiFi (upper medium)<br>- WiFi (high) |
| *Wearables* | | + | + | +- | BAN/PAN | p2p, mesh | - | Low | BLE, ZigBee, Z-Wave, Bluetooth |

Table 3.5: Recommendations for selecting wireless technology for smart home applications

**Recommendations**

We analyzed these technologies in order to determine their suitability for smart home applications with regards to the power, cost, security, range and scalability, and throughput requirements. In Table 3.5 you can see our recommendations about wireless technologies sorted by their suitability for different smart home applications. For example, ZigBee and BLE are most appropriate for home lighting, heating and ventilation systems, and wearables. For home security systems with door and surveillance cameras and for entertainment devices Bluetooth and WiFi can be used, and only WiFi when we have high throughput requirements (streaming a real-time video or file transferring).

For custom home automation projects, other simple, cheap, and low-power wireless communication modules based on custom technologies can also be used. Their characteristics may vary widely. For example, with Arduino you can use XBee (IEEE 802.15.4), RF 433 MHz, RF 2.4 GHz, WiFi (IEEE 802.11), Bluetooth/BLE (IEEE 802.15.1) modules.

#### 3.4.2.2 Home Automation and Smart Energy profiles of ZigBee

The IEEE 802.15.4 standard was designed by Institute of Electrical and Electronics Engineers (IEEE) to support a healthy trade-off between energy consumption, data rate, and range for low-rate wireless personal area networks (LR-WPANs) with a focus on very limited battery consumption requirements [8, 59] because of devices such as sensors and embedded devices with low energy availability, constrained memory, and processing capabilities [15]. Original standard from 2006 was updated few times to support new physical layers and recently opened frequency bounds in China and Japan. It can operate on three frequency bands: 868 MHz (Europe), 915 MHz (USA, Australia), and 2.4 GHz (worldwide) in a short-range of around 10 meters with transfer rates respectively of 20, 40, and 250 Kb/s. It supports next capabilities: addressing, beaconing and optional usage of guaranteed time slots (GTSs), Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) or ALOHA channel access protocols, energy detection and link quality indication, and acknowledgment. There are two methods for channel access: beacon and non-beacon enabled. When beaconing is used a time between beacons is divided into three intervals: a contention access period with CSMA/CA; a contention-free period with allocated GTSs for each node; an inactive period. During the inactive period, nodes may sleep. If no beaconing is used, devices use plain CSMA/CA scheme [19]. The standard defines two types of devices: a full-function device (FFD) and reduced-function device (RFD) according to their role in the network and their resources. Based on the standard two network topologies can be used: star topology and peer-to-peer topology. Also, combinations of these like cluster tree and mesh of multiple clusters can be built. Every network must include at least one FFD which acts like a personal area network (PAN) coordinator [59]. Physical data frames have at most 127 bytes. Such packets are small to minimize the probability of errors [8, 18, 59].

IEEE 802.15.4 is the basis for the ZigBee, ISA100.11a, WirelessHART, MiWi, and Thread specifications.

ZigBee standard enhances the IEEE 802.15.4 standard by adding network and security layers and application framework [60]. An open, non-profit association called ZigBee Alliance established in 2002 contains over 400 members from around the world and is responsible for the development of the technology and maintaining and publishing ZigBee standards. The main purpose of the ZigBee Alliance is to provide interoperable technology and certify ZigBee compatible products to ensure that they work together seamlessly, even if they’re from different companies [50]. They developed standard application profiles for different manufacturers such as Smart Energy [62], Home Automation [61], Telecom Services, Health Care, Building Automation, Remote Control, Input Devices, Light, and Retail Services. Three of them are relevant to smart home.

*Home Automation profile* (current version is ZigBee Home Automation 1.2) is based on ZigBee PRO network and includes devices like home appliances, thermostats, door locks, security sensors, lights, etc. ZigBee PRO network specification allows star, tree, and mesh topologies and can be used to build large networks of low-power devices. Mesh network topology helps to increase the range and reliability of the network with the possibility of full peer-to-peer communication. ZigBee PRO specification also includes an optional feature called Green Power which allows connecting to the network self-powered and energy harvesting devices [60]. Such devices usually work with sleep cycles and poll control technology helps to reduce their power consumption (100-500 $\mu$J for one communication). A lifetime of battery-powered devices can be increased up to 7 years [61]. ZigBee PRO networks have three device types: ZigBee coordinator, ZigBee router, and ZigBee end device [60]. Coordinator (PAN coordinator) is the main device in the network. It must be a full-function device. PAN coordinator is responsible for starting the network, initiating and maintaining the devices on the network, storing information about the network and security keys, acting as a trust center. The network can contain up to 255 members, where one is the coordinator. PAN coordinators are usually mains-powered. A router is capable of routing messages between other devices and supporting associations according to its hierarchical routing strategy. It can be used to extend the network. A router in distributed network architecture can also be used like trust center to distribute keys within the network. Routers are usually mains-powered. Other devices which are neither a coordinator nor a router (can be RFD or FFD) with specific sensing or control functions perform like end devices. RFD can't communicate with other RFD but only through a router or coordinator. End devices are usually simple battery-powered devices (e.g., light switch, security sensor). The feature of ZigBee PRO network that allows nodes to send a request to the coordinator for changing frequencies due to interference is called frequency agility. This can increase the network performance only if the request gets through. To connect ZigBee PRO network to the Internet optional combination of coordinator/gateway may be used [60].

*Smart Energy profile* is designed for metering and energy management. It can be used to monitor, control and automate usage of energy and water to reduce their consumption and save money. Smart Energy profile supports ZigBee and ZigBee PRO nodes. It has higher security requirements on the network than in Home Automation, so all the devices in the network must support installation and use of link keys. Also, these two networks usually are not connected [62]. ZigBee IP specification was designed to bring IPv6 connectivity to low-power, low-cost devices of ZigBee networks. In particular, it was developed to support ZigBee 2030.5 profile (a successor of ZigBee Smart Energy 1.x profile also known as Smart Energy 2). Support of full IPv6 network stack allows achieving scalability and interoperability with other devices and networks on the Internet without the need of intermediate gateway or protocol translation. Thus, ZigBee IP can use such protocols like 6LoWPAN, RPL, TCP, UDP, TLS, PANA, and mDNS. But no other Zigbee standards are capable of operating on an IP network and so cannot utilize this specification [63]. ZigBee IP stack plus SEP 2 ESI (electronically stored information) with full function set need at least 220 Kb flash and 12 Kb RAM [62]. ZigBee IP networks have 3 types of devices: ZigBee IP

coordinator, ZigBee IP router, and ZigBee IP host. A border router can be used to connect ZigBee IP network to the Internet.

Finally, ZigBee RF4CE specification offers a simple device-to-device communication for remote control without full-featured mesh networking capabilities [19].

Commissioning of Zigbee devices is very easy. They support self-configuring deployment without direct human involvement (A-mode). Also, deployment can be done by the end user or installer with stimulus and feedback mechanisms from devices (E-mode). It also supports pairing between two devices (like light bulb and switch) without a centralized coordinator. In the last mode, a ZigBee commissioning is done using external tools to facilitate more extensive commissioning (S-mode). ZigBee network contains mechanisms for self-checking and self-curing. For increasing reliability, interoperability and flexibility ZigBee devices have remote firmware update option.

**Security**

The latest version of ZigBee is the recently launched 3.0, which is essentially the unification of the various ZigBee wireless standards into a single standard. ZigBee Specification includes security provisions and options, that improve the basic security framework defined in IEEE 802.15.4, focusing also on key establishment and distribution for packet encryption. The IEEE 802.15.4 provides security using Advanced Encryption Standard (128-bit AES) with different security modes (no security, AES-CBC-MAC, AES-CTR, AES-CCM) to provide confidentiality, data authenticity and integrity, also semantic security and protection against reply attacks are provided (frame counter and using different nonce or initialization vector) [8]. Message Authentication Code (MAC) is used for data integrity. MAC can have a size of 32, 64, or 128 bytes depending on selected level of security.

Packet security in ZigBee PRO networks is available at two levels – network-level security using a key shared by all nodes in the network (network key) and application-level security using a link key shared by a pair of nodes for end-to-end communication (application key). Application key is a master key or a link key transported by the trust center. Master key may be pre-installed in each node. The function of the master key is to secure the key establishment procedure. Trust center master key must be unique for each device on the network. Each device has a table called Access Control List (ACL) to determine which devices are authorized to perform a specific function. This table may also store the security material (keys, security modes, frame counters) used to secure communication with other devices [64]. ZigBee includes network re-keying (using key transport or pre-installation) and key negotiation protocols [60]. Network key must be changed frequently. The new network key is shared using the old network key. A new device can join the network only if it has the network key. There are two security policies for the trust center:

- *commercial mode*: the trust center shares both master and link keys with any of the devices in the network;
- *residential mode*: the trust center shares only the network key.

Commercial mode requires more memory resources but provides a high level of security. The residential mode is often used for networks with constrained devices like WSNs [64].

ZigBee IP specification uses PANA protocol for key distribution. It also supports ECC public key infrastructure and X.509 v3 certificates, PANA/EAP based network authentication and admission control. Link-layer security for frame protection is based on the AES-128-CCM algorithm. TLS1.2 protocol can be used for end-to-end security [63]. ZigBee Specification provides two security models for standard and high-security applications (Standard Security Mode and High Security Mode). The security model of the Smart Energy Profile is a trade-off between the two standard models. But these security models have some critical issues [65]. For example, upon leaving the network, the key of the node is still can be used for accessing communication, it is not properly revoked. The second problem is the scalability issue in the public-key protocol because of limited storage resources of the end devices. As it was shown many times by different researchers in practice security of ZigBee and many other similar home automation technologies is quite poor because of constraints of the home devices, choice of interoperability and usability over security. Not always security is developed by security people, and sometimes they try to develop their own security algorithms what is risky. It may be very easy for the attacker to deactivate your security or take control over your home devices. In some situations, it can be useful to develop custom solutions for home automation applications using well known and proven by time security mechanisms. But only with a clear understanding how your application works user confidence is increasing.

# 4 METHODOLOGY AND TECHNOLOGY

## 4.1 Methods

In Chapter 1 we defined a problem statement and main objectives to present the arguments and achieve the main aims of the research. The next step is defining a methodology which will help reach the objectives of this research paper. We will present it in Table 4.1 by mapping the research questions to the aims, and methods selected to find answers for these questions.

| *Number* | *Aims* | | *Research questions* | | *Methods* | |
|---|---|---|---|---|---|---|
| 1 | Determine which communication technology is most suitable for data transferring in low-power, low-bandwidth Internet of Things networks | A1 | • What are the requirements for smart home applications?<br>• What communication technologies are used in low-power, low-bandwidth smart home networks? | Q1<br>Q2 | • Literature review<br>• Comparison and analysis | M1<br>M2 |
| 2 | Determine how does security affect the power consumption of constrained IoT device and performance of the wireless communication channel | A2 | • How to build a home automation project based on cheap and easy to use Arduino microcontrollers?<br>• How does security affect the performance and power consumption of constrained Arduino microcontroller? | Q3<br>Q4 | • Technology study<br>• Design and modeling<br>• Simulation and measurements<br>• Analysis of the results | M3<br>M4<br>M5<br>M6 |

Table 4.1: Mapping of aims to research questions and research methodology

As you can see we will use following methods:

*Literature review* to define common applications and communication technologies used in a smart home scenario and requirements for wireless communication technologies. We analyzed background and related works to the current state of the art of the Internet of Things and smart home; related to the comparison of low-power, low-bandwidth home wireless communication technologies; related to the security, performance, and power consumption of constrained IoT devices and Arduino microcontrollers.

*Comparison and analysis* of ZigBee, WiFi HaLow, Bluetooth, BLE, ANT, and Z-Wave wireless communication technologies in terms of performance, power consumption, and security in order to determine most suitable technologies for smart home applications (home automation, energy management, entertainment, wearables).

*Technology study* of Arduino microcontrollers (Arduino Nano and Arduino Mega) and their characteristics, features of the equipment; Arduino IDE and features of the programming; RF

433 MHz wireless communication modules (transmitter and receiver) for Arduino and their characteristics; AESLib and VirtualWire libraries for Arduino.

*Design and modeling* of a battery-powered sensor node based on Arduino with sleep management in home automation scenario; of experiments on Arduino to determine execution time and power consumption of the sensor node.

*Simulation and measurements* of power consumption and execution time using the aforementioned hardware and software, and also laboratory equipment (microammeter, digital multimeter, etc.) when sensor node is transmitting data with different levels of security (without encryption; AES-128, AES-192, AES-256; AES-128, AES-192, AES-256 with counter).

*Analysis of the results* to investigate a correlation between security, power consumption, and performance of sensor node and give practical recommendations for developers of home automation projects.

## 4.2 Technologies

Depending on the type of IoT application and its goals different platforms may be chosen. For home automation system and other DIY projects most popular two platforms: Arduino and Raspberry Pi. In this subsection, we will compare them to select appropriate device for building a home automation project and for experiments. It should be cheap, easy to use and program, have enough resources for simple home automation applications, and consume little electricity.

**Arduino**

Arduino is an open-source and easy-to-use electronics platform based on hardware and software. It's also a company and a user community. Arduino is designed to build digital devices and interactive objects that can sense and control objects in the physical world. It includes a physical programmable circuit board and an integrated development environment (IDE) which is used to write and upload programs by sending a set of instructions to the microcontroller from the personal computer. Usually, USB cable is used for this. Arduino programming language is a dialect of features from the C and C++. It's based on Wiring and Arduino IDE is based on Processing [66].

Arduino was born in an educational environment at the Ivrea Interaction Design Institute, Italy, aiming to provide their students with a low-cost and easy tool for fast prototyping. Arduino became quite popular with people just starting out with electronics and without any background in programming.

It's important to realize what Arduino is a microcontroller, not a full fledged computer. It doesn't have an operating system and can't do multitask operations, have very small amount of memory, usually 2 Kb of RAM and 32 Kb of flash memory (see Table 4.2), and simple processor. As a result, it can run only small, highly focused applications [67]. Only a small amount of data can be saved in EEPROM.

Various expansion boards and communication modules like Ethernet, WiFi, and XBee shields can be attached to the Arduino.

### Raspberry Pi

A Raspberry Pi is a tiny low-cost single-board computer with a size of credit card. It was developed in the United Kingdom by the Raspberry Pi Foundation. The goal was to create a device with small size and accessible price that would improve the programming skills and hardware understanding of the students. Typically the Raspberry Pi runs some version of the Linux operating system. The Raspberry Pi is slower than a modern laptop or personal computer but is still a complete Linux computer with a possibility to connect mice and keyboard. Most Raspberry Pi models have at least 512 Mb of RAM and SD card for permanent storage of OS and data. It can multitask processes and run multiple programs in the background.

The Raspberry Pi Foundation promotes Python and Scratch as main programming languages but the flexibility of this platform lets you use other languages such as Ruby, PHP, and Java.

The majority of Raspberry Pi boards come with an embedded Ethernet connector, some of them are equipped with WiFi and Bluetooth. There is also a possibility to attach a WiFi dongle.

### Comparison

But Raspberry Pi like any other computer requires some time to boot up and needs to be shut down properly [67]. Raspberry Pi doesn't have embedded analog pins, only digital. It doesn't have such simplicity and reliability. Although the price difference between Arduino and Raspberry Pi is usually not so significant, what is important that power requirements of Raspberry Pi are much larger. Input voltage is not so flexible like in Arduino. The power consumption of Raspberry Pi Zero is at least 160 mA. Other models are more resource intensive. Despite significant compute capabilities of Raspberry Pi, a simplicity of Arduino makes it much better for pure hardware applications like sensor reading or motor driving [68].

Hence, a combination of these platforms in one project may be a solution for big IoT applications. But for small constrained nodes with cheap communication and simple function to read sensor data and transmit it, Arduino microcontroller will serve the best.

There are plenty Arduino boards to choose [66]. All of the boards in Table 4.2 are based on 8-bit Atmel microcontrollers and are quite constrained devices. For example, small boards like Arduino Gemma may be used for wearables. They have just a few pins and very limited memory (512 bytes of RAM). But they consume much less energy, what can be quite important when they work on tiny batteries. Arduino Nano is a compact board with a more powerful microprocessor, more memory and more digital and analog pins to connect sensors and other devices to the board. According to the classification presented in Table 3.2 both devices have class C0. The third board from the table is Arduino Mega. It is much larger. The main advantage of this board, that it is designed for much complex projects. It has quite a lot of pins and it can store more data in memory (8 Kb) and have more place for programs (256 Kb). It can also store data in 4 Kb non-volatile electrically erasable programmable read-

only memory (EEPROM). It can be useful for data logging. So, we can say that it has class C1. There is even a possibility to connect Ethernet shield to your Arduino Mega. It will support TCP and UDP for up to 8 simultaneous connections with limited buffer size.

| *Board* | *Arduino Gemma* | *Arduino Nano* | *Arduino Mega 2560* |
|---|---|---|---|
| *Microcontroller* | ATtiny85 | ATmega328 | ATmega2560 |
| *Size* | 27.94 mm (diameter) | 18 mm x 45 mm | 53.3 mm x 101.52 mm |
| | Small / wearable | Medium / portable | Medium / portable |
| *Weight* | 3 g | 7 g | 37 g |
| *CPU frequency* | 8 MHz | 16 MHz | 16 MHz |
| *Type of microcontroller* | 8-bit | 8-bit | 8-bit |
| *Input voltage* | 4-16V | 7-12V (6-20V), 5V | 7-12V (6-20V), 5V |
| *Power consumption* | 9mA | 19mA | 25mA (9V) |
| *RAM* | 512 bytes | 2 KB | 8 KB |
| *ROM / Flash* | 8 KB (-2.75 KB) | 32 KB (-2 KB) | 256 KB (-8 KB) |
| *EEPROM* | 0.5 KB | 1 KB | 4 KB |
| *Digital pins* | 3 | 14 | 54 |
| *Analog pins* | 1 | 8 | 16 |
| *Cost* | 15$ | 19$ (2.7$) | 40$ (8$) |

Table 4.2: Comparison of different Arduino boards

We will use for testing purposes Arduino Nano with RF transmitter module as a sensor node and Arduino Mega with RF receiver as a control node in HAN. Arduino Mega has more computational capabilities to process the data, store data in the memory and send it to the Internet. But 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 MX-RM-5V work on unlicensed frequency 433.92 MHz in LPD433 (Low Power Device) band [69]. 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. They can only send data in one direction (simplex mode), so there is no acknowledgment. For bidirectional communication (half duplex mode) you should use two pairs of them.

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). ASK is a type of amplitude modulation that represents digital data as a finite number of amplitudes, each assigned a unique pattern of binary values. In this case, a binary one is represented by the presence of a carrier wave and binary zero by its absence (on-off keying). Amplitude modulation is sensitive to noise and distortion. 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.

*Transmitter FS1000A specifications:*

- Output power: up to 40 mW
- Supply voltage: 3-12 V
- Current consumption in transmission mode: 20-30 mA
- Current consumption in standby mode: 0 mA
- Size: 19 x19 x 8 mm
- Weight: 2 g

*Receiver MX-RM-5V specifications:*

- Supply voltage: 5 V
- Current consumption: 4 mA
- Size: 30 x 14 x 17 mm
- Weight: 4 g

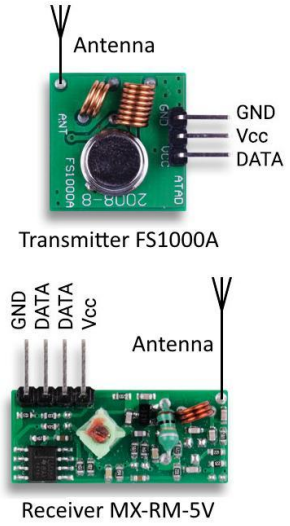



Figure 4.1: Low-cost RF 433 MHz modules [69]

The price for a pair may be around 1$.

We are using two external coil antennas with a length of the quarter of the wavelength ($\lambda/4$) for both transmitter and receiver. That will allow us to use a lower power supply at the transmitter.

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*[70] 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 [70].

**RF 2.4 GHz wireless communication modules**

Another cheap ultra low power wireless module for Arduino is nRF24L01+. It operates in worldwide 2.4 GHz ISM band and allows data rates 250 kbps, 1 Mbps and 2 Mbps. nRF24L01+ uses GFSK modulation and has 126 channels. Maximum payload size is 32 bytes. Unlike RF 433 MHz, these modules are transceivers, and by default, they use auto

acknowledgment and auto retransmission. Peak currents are lower than 14 mA. Supply voltage from 1.9 to 3.6 V. Each transceiver has printed circuit board (PCB) antenna. The range is close to 250 meters light-of-sight.

The price for a transceiver is around 0.65-1$.

**Comparison**

RF 433 MHz modules were selected for further experiments because they are easier to configure and they use much fewer pins to connect to the board, don't have some power problems like RF 2.4 GHz, you can power them directly from Arduino.

Both modules don't use any encryption, so security, if needed, must be implemented in the application layer (i.e., in Arduino sketch).

## 4.3 Design of experiments

In the practical part of this thesis we will consider battery operated and/or energy harvesting constrained devices which usually use normally-off or low-power operating strategies to collect and transmit sensor data over short-range simplex wireless communication (see Table 3.1). For an example, we are considering sensors in the home automation system. Battery-powered sensor devices like temperature and humidity sensors transmit data periodically and their transmission frequency is relatively low due to the slow change of parameters being monitored. Other sensors may transmit data based on events like light sensor or motion detector.

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 4.2. In our case, there are three sensors. The sensor node is powered by a battery. To save power it goes to sleep between measurements. There is no need in most cases to have bidirectional connection with the sensor node, so we can save money using just one transmitter. In that case, there is no reason to constantly listen to the channel for incoming messages. But communication will not be reliable (no acknowledgments or retransmissions). We will not be able to configure sensor node remotely. From the other side, since there is no need to wait for acknowledgment, therefore it can go into sleep mode just after the data transmission. The coordinator is usually mains-powered with a permanent connection to the Internet. Ethernet shield for Arduino can be used for this. The web server of home automation system may be based on Raspberry Pi and contain web interface with a home automation management system for monitoring and controlling the system. The coordinator is responsible for sending commands between devices automatically based on a set of user rules and data from sensors or on demand.

In this research, we will only consider link-layer unidirectional communication and link-layer symmetric encryption between the sensor node and coordinator as depicted in Figure 4.2. All the measurements also are done on the sensor node.

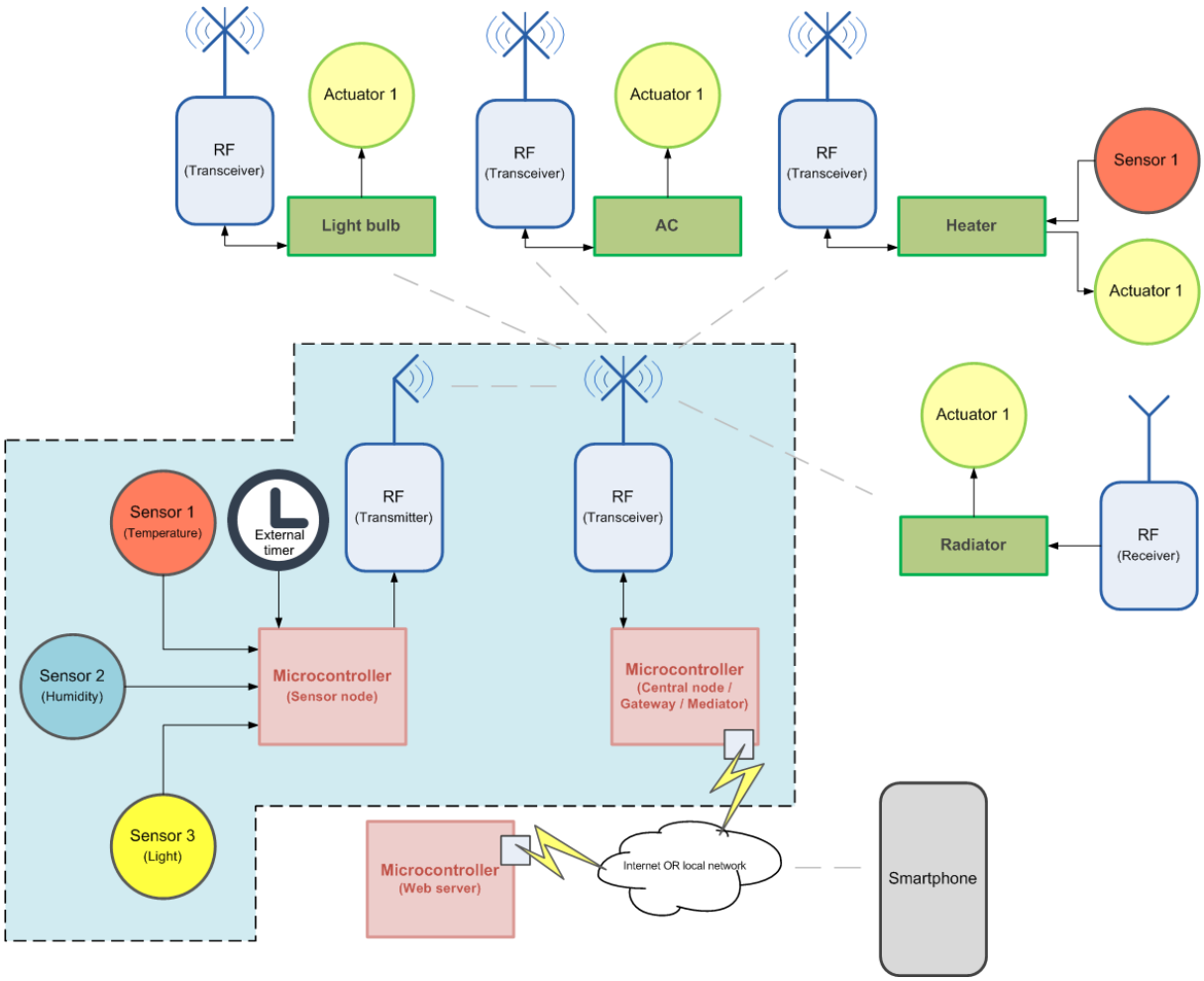



Figure 4.2: Home automation model with a sensor node

There are a few parameters of such a device which can be used for adjusting the power consumption and performance of the device:

- periodicity of sensor data readings;
- periodicity of transmitting data;
- preprocessing of data;
- size of the data send by one period;
- processor frequency and type of processor;
- transmitter frequency, modulation, encoding, error handling algorithms, and communication protocol;
- acknowledgement and number of retransmissions;

- transmitter power;
- type of transmitter's antenna and its size;
- communication speed (bit rate);
- sleep mode;
- encryption and other cryptographic algorithms.

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. Calculating some average results can decrease the amount of data we need to send and transmission periodicity. If we send fewer data smaller package will be sent and there will be less probability of packet corruption. But when we need to send more data, few packages will be sent, increasing the data overhead and time of transmission. 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 not save any sensor data to the memory storage and will not process it. Furthermore, we will not consider reading the data from the sensor in the experiments, it will be generated randomly. The *VirtualWire* library doesn't allow sending a package with payload size larger than 27 bytes and doesn't include information about message recipient and/or sender so that information if necessary can be added directly to the payload. 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.

It will be interesting to determine the impact of security in data transmission. A lot of communication technologies in IoT base their security on AES symmetric encryption algorithm because it's a well-known cipher with reliable security which is used even in banking institutions. We will use it as a reference algorithm. Block size of AES is 128 bit, so the main disadvantage when using it for small messages in IoT is associated with the need to extend the message to the block size, so the result message will be increased in size to 16 bytes no matter how small it was before. More data we send, more resources we need. So it will be logical to increase the message size with useful data. As an example, it can be a MAC, CRC, counter, etc. The next disadvantage when using security is a need to store the key. For AES the smallest key is 128 bits. It can also be 192 or 256 bits for higher security level. Also, encryption algorithm takes a place in the program, uses some variables and buffers in memory, it takes CPU time and increases the communication delay.

In the next section, we will try to measure the impact of security on constrained IoT devices.

# 5 EXPERIMENTS

To investigate the correlation between security, power consumption, and performance of a sensor node based on Arduino Nano a few experiments were conducted in order to measure execution time and power consumption of the device when transmitting data over wireless communication with different levels of security.

To present arguments we need to carry out at least the following measurements:

- *Experiment 1*: Measure the execution time of AES algorithm on Arduino microcontroller.
- *Experiment 2*: Measure the time of data encryption and transmission using RF 433 MHz wireless communication module for Arduino Nano when using different levels of security based on AES encryption.
- *Experiment 3*: Evaluation of power consumption of Arduino Nano microcontroller when using different levels of security during the data transmission.

## 5.1 Experiments setup

For the experiments we will need following equipment:

- Arduino Nano;
- Mini-B USB for connection the Arduino Nano to the PC and powering it with 5 V;
- RF 433 MHz transmitter FS1000A and receiver MX-RM-5V;
- Arduino Mega 2560;
- USB for connection the Arduino Mega to the PC and powering it with 5 V;
- microammeter;
- digital multimeter;
- 5 V DC external battery;
- PC;
- some wires;
- some resistors.

We also will need the next software:

- Arduino IDE 1.6.0 on a PC for uploading the sketches to the microcontroller and reading messages from the hardware serial communication via Serial Monitor.

Time measurements in this paper were done using standard Arduino function *micros()* from the sketch. During the experiment 1, a simple sketch which uses AES library for Arduino for encryption and decryption of a block of a random data or a few blocks in CBC mode was

uploaded to the Arduino Mega and Nano through a USB. Measurements were repeated with keys of different sizes and for different numbers of blocks. Results were shown to the user via Serial Monitor on the PC. These results were the starting point of performance evaluation of the device.

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:

- using an internal timer;
- using the watchdog timer;
- using the hardware UART (serial interface over USB);
- 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 (see Subsection 5.2 below about different modes) 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 5.1.

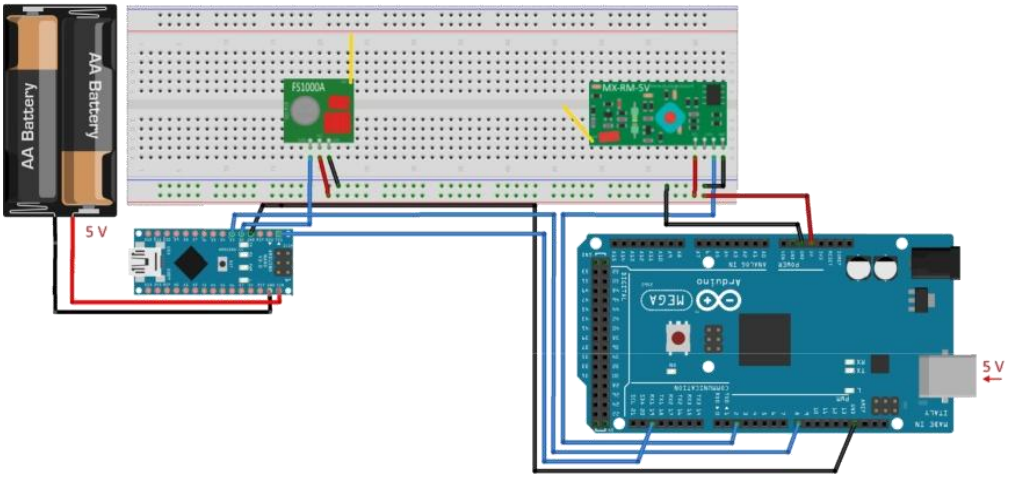


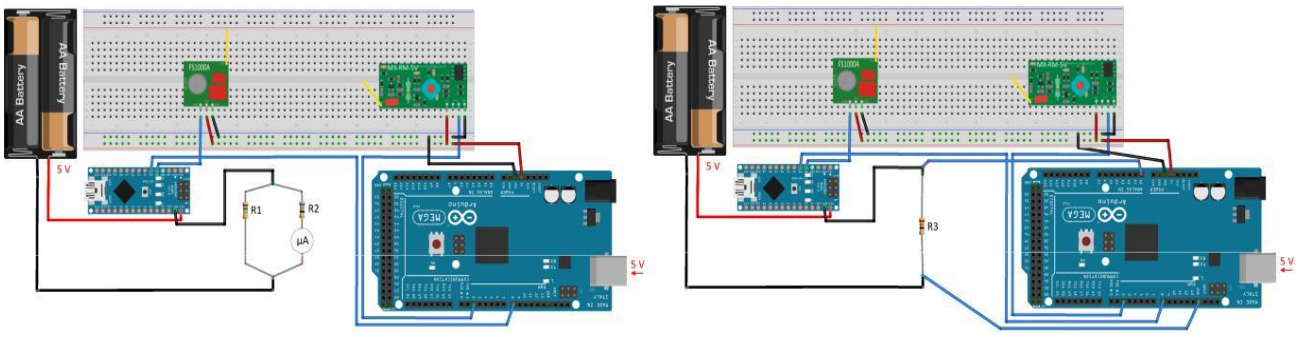

Figure 5.1: Setup of experiment 2

There are a few possibilities to power the Arduino Nano microcontroller: via the Mini-B USB connection, via unregulated or regulated external power supply. We will use Mini-B USB or external 5 V battery for powering the device depending on the situation. Arduino Nano also has regulated 3.3 V and 5 V output pins which can be used for supplying the RF transmitter.

For controlling the sensor node during the experiments and as a tool of measurement we will use another more powerful Arduino – Mega 2560. Arduino Mega will be connected to the PC via USB cable all the time and powered from it. 3.3 V and 5 V output pins of Arduino Mega can be used like a power supply for external devices, for example, RF 433 MHz FS1000A transmitter and only 5 V for powering the MX-RM-5V receiver. One of the data pins of the receiver is connected to the pin 2 of the Arduino Mega.

During the experiment 2 measurement results of encryption and transmission time were transmitted after the end of the experiment from the Arduino Nano (sensor node) to the Arduino Mega (controller) through the hardware serial port of Arduino Nano to one of the serial ports of Arduino Mega. Results were shown to the user via Serial Monitor on the PC. Measurements were repeated for different amounts of data and with different data transfer rates of the communication module.

Power consumption measurements were divided into two parts. During the first part, a current consumption of Arduino Nano from the battery was measured using microammeter. With a system of two resistors and a microammeter current divider was built allowing current measurement with microammeter on one of the branches (see Figure 5.2.a). During the second part of the experiment, voltage measurements from the power line were made using an analog pin of Arduino Mega controller over the shunt resistor (see Figure 5.2.b). After that current was calculated using Ohm's law. After calibrating Arduino Mega was taking samples from analog input frequently (5 times per millisecond), displaying the results via Serial Monitor after each period and calculating the average results of current over the time period. The voltage measurements and control current measurements were done using a digital multimeter. Results of measurements were compared.




(a) using microammeter (b) using Arduino Mega

Figure 5.2: Setup of experiment 3

## 5.2 Parameters studies

To measure the lifetime of IoT device we should consider its duty cycle and power consumption in different stages of its activity. There are three main stages of device activity [28]:

- inactive stage;
- start-up and shut-down stage;
- application runtime.

It may work on a schedule or wait for an external event. During the inactive stage, a device can be in different modes of sleeping/awakeness. It can be in deep sleep mode when only waiting for a signal or event to wake it up, in idle mode when almost all components of the device are still working, and in waiting mode. Therefore, devices in different modes consume different amounts of energy but usually less than during normal operation. After inactive stage device needs some time to wake up.

Arduino microcontrollers as other 8-bit microcontrollers from AVR family support some of next sleep modes:

- SLEEP_MODE_IDLE;
- SLEEP_MODE_ADC;
- SLEEP_MODE_EXT_STANDBY;
- SLEEP_MODE_PWR_SAVE;
- SLEEP_MODE_STANDBY;
- SLEEP_MODE_PWR_DOWN.

SLEEP_MODE_PWR_DOWN provides the most power savings, SLEEP_MODE_IDLE is the least power saving mode but the most functional. Arduino Nano (ATmega328) and Arduino Mega (ATmega2560) support all of them. For Gemma (ATtiny85) only 4 modes are available including SLEEP_MODE_IDLE and SLEEP_MODE_PWR_DOWN. A power consumption of the microcontroller during inactive stage and start-up stage is more or less given by the documentation regardless of the application [28]. For example, datasheet of ATmega328 contains information on electricity consumption of about 0.2 mA in active mode, 0.1 $\mu$A in power-down mode and 0.75 $\mu$A in power save mode at 1 MHz.

But there are a lot of extra hardware in Arduino boards which consume power, for example, power regulator can significantly increase the power consumption of the device even in power-down sleep mode. The other very power intensive component of the board is FTDI USB-to-TTL serial chip. So, the power consumption of Arduino Nano board according to [66] will be around 19 mA. In practice, that value can be significantly higher depending on the application and configurations of the board. To reduce power sleep mode must be as deep as possible and all unused components of the device must be disabled by software:

1. Floating output pins.

2. Internal timers.
3. Watchdog timer.
4. Brown-out detection.
5. Analog-to-digital converter (ADC).

It's also possible to lower the input voltage of the device and try to reduce the clock speed [27]. Minimizing idle listening, collisions, and protocol overhead will help to reduce power usage of communication module [24]. Synchronization of nodes and duty cycling are used to increase the lifetime of the node.

Hence, in some cases, user application will be the most power intensive part of the program but for battery-powered devices sleep mode is crucial. So, for increasing lifetime of the device, it's quite important to optimize the program in accordance with the electricity consumption. It's expected that more significant impact will have sleep mode efficiency and the duration of the program, type of performed operations. Consumed energy of the device will define the battery life and is reflected by the area under the curve of consumed power during the duty cycle of the device, as shown in Figure 5.3:

$$E = \int_{t_{start}}^{t_{end}} P(t)dt. \tag{5.1}$$

And electrical power in every point is equal to the product of voltage and current: P = UI. Watt (W) is a unit of power and energy is measured in Joules (J) or in "watt-hours" (Wh). The relation between Joule and "watt-hour" is next: 1W h = 3.6 kJ. The start-up time and shut-down time are depicted by s1 and s2, application runtime by t1 and sleep time by t2. The duty cycle of the device will be determined by the division of working time t3 by total time period T in percentage. Duty cycle scheduling aims to minimize the sleeping delay, and as a result, latency and power consumption.

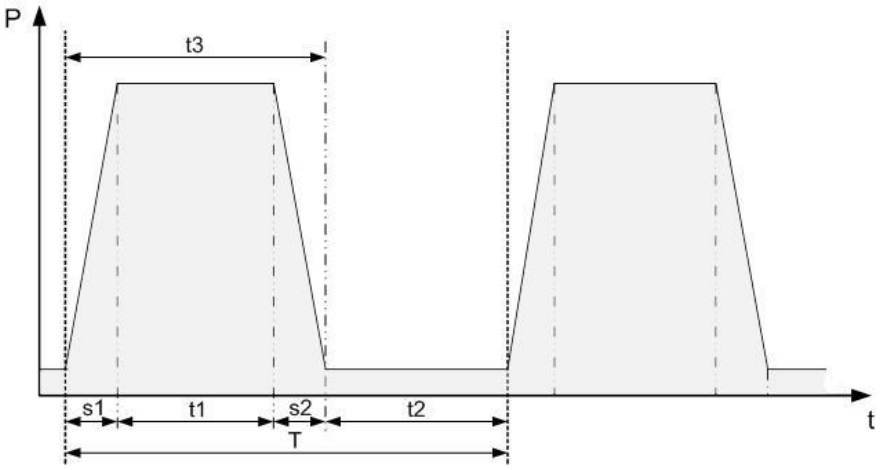



Figure 5.3: Duty cycle and energy consumption of a battery-powered device

Wireless communication is quite expensive operation for constrained devices. For comparison, one bit of information transmitted through the RF communication channel corresponds to executing 800-1000 instructions by power consumption [15].

During the experiments we will investigate/change next input parameters:

- *payload data size*: 4 bytes, 8 bytes, 14 bytes;
- *encryption*:
  - without encryption;
  - AES-128, AES-192, AES-256;
  - AES-128, AES-192, AES-256 with counter (2 bytes);
- *sleep management:*
  - without sleeping;
  - IDLE, ADC, EXT_STANDBY, PWR_SAVE, STANDBY, PWR_DOWN;
- *communication data rate:* 1000 bps, 3000 bps, 6000 bps;
- *RF power:* 3.3 V, 5 V;
- *RF power strategy:*
  - from Arduino;
  - from individual power supply.

Where the counter is a fixed-sized data inserted before the plaintext. The counter is changing for every packet, usually increasing by one. To ensure security each combination "key + counter" should be unique.

## 5.3 Discussion of results

So, if you decide to add encryption to the sensor node what will be the influence on the performance of the device? First of all, let's consider the overhead of encryption time.

**Encryption**

As a result of experiment 1, we got a performance comparison of AES algorithm based on the key length and number of encrypted blocks in AES-CBC mode on Arduino Mega microcontroller using AESLib[71] library for Arduino. Results showed that it takes approximately 0.27 ms for Arduino to encrypt one block of random data and 0.35 ms to decrypt it when it doesn't do any other tasks (see Figure 5.4). With increasing of a key size, the time of the algorithm is increasing too and not linearly. It will be interesting to compare the execution time in a real application to determine the impact of CPU load on the execution time of encryption algorithm for a small amount of data. Also, results regarding the CBC mode of the algorithm for different amounts of blocks are obtained. The trend is clear. The same results were obtained on Arduino Nano. The measurement error is in average around 3

microseconds above on each measurement using the micros() function of the Arduino. Average results of 250 iterations were calculated.

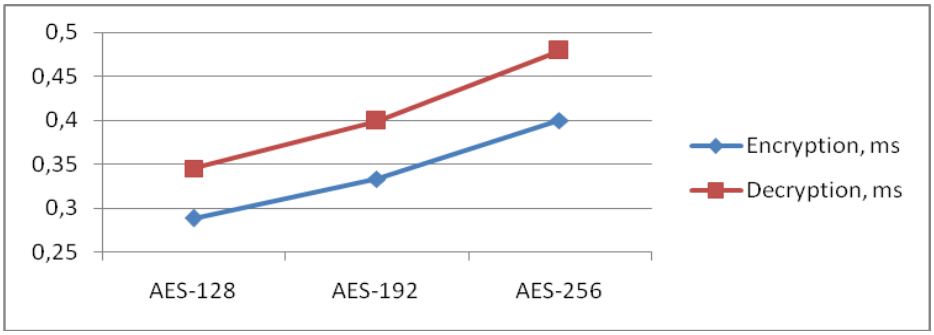



Figure 5.4: Execution time of encryption and decryption of one block of AES algorithm in Arduino

Comparing these results with results from the second experiment of execution time on our sensor node we can see that the execution time of the algorithms depends on the transmission rate of the transmitter as shown in Figure 5.5. That can be explained by more intensive usage of the Timer1 of Arduino by the *VirtualWire* library on increased frequency even if even if there is no data transmission in that moment. It reduces the performance of the microcontroller. It's expected that execution time of the algorithm when using counter will be the same as in usual mode. The only difference can be in the smaller amount of user data we can add to the block. In our case counter size is 2 bytes. After overflowing the counter, and it's once per 65535 usages, the key should be changed. But when user data is less than 14 bytes like in our sensor there should be no difference only increasing the security by protection from the replay attack which is one of the most common in wireless communication. But as shown in Figure 5.6 the encryption of AES-256 with counter will take more time than usual AES-256 when using high communication speed of the RF module.

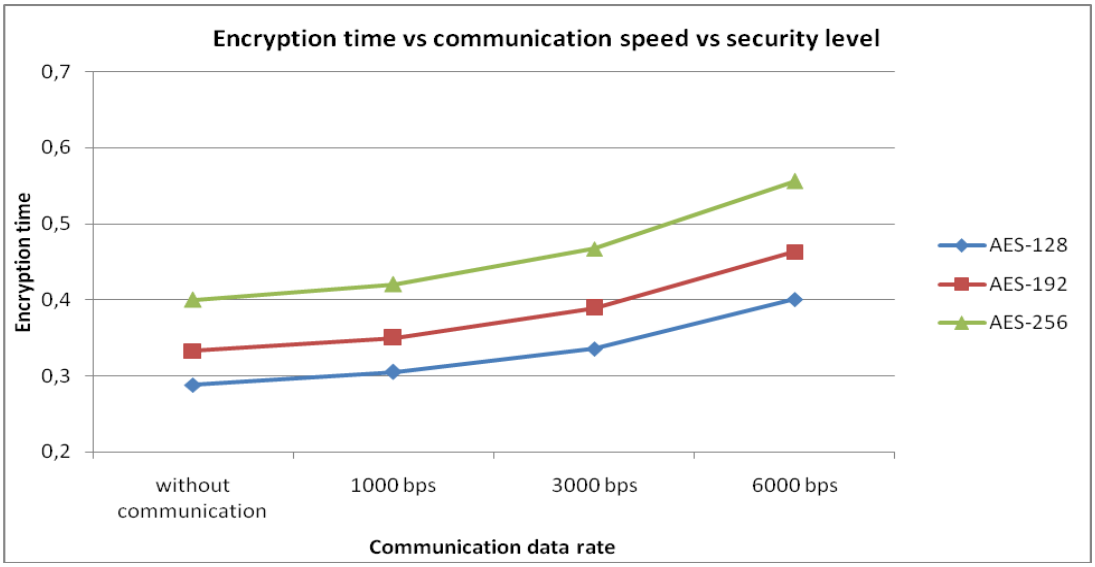



Figure 5.5: Relationship between encryption time and transmission rate of wireless module

When using encryption we should expand our payload to the size of 16 bytes and then send 16 bytes instead of, for example, 4 bytes of temperature data.

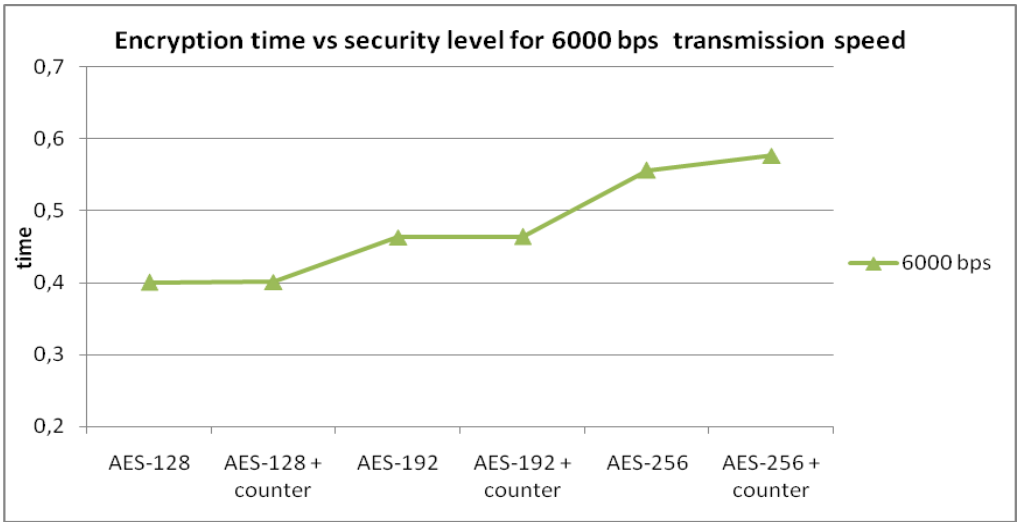



Figure 5.6: Increasing of the encryption time for 6000 bps transmission speed

**Transmission**

As you can see from Figure 5.7 it would take around twice more time to send encrypted sensor data than when sending a plain text and even 2.5 times more for a data rate of 6000 bps. It should be also mentioned that sending one block of data at speed 3000 bps takes the same time like encrypting of about 270 blocks with AES-128 and almost 300 blocks with AES-256 comparing results with Figure 5.5 but during that time Arduino also can continue running its program. So, most important for our sensor node will be the amount of data we send and if we use encryption the block size of the used encryption algorithm. For example, if the developers of home automation system will use some encryption algorithms with smaller block size, for example, 64 bit like in DES, 3DES, IDEA or lightweight PRESENT, TEA, RC5, or HIGHT which are also popular in the Internet of Things, the transmission time can be reduced by 30 percent. Smaller block size will not be a weakening of security if we change key frequently enough. When using counter, counter size must define the rekeying interval. There are few possibilities how can unidirectional nodes change their keys. A new key can be calculated on time or event basis from the old one and some secret information shared between the nodes or based on the time synchronization and on a common key generation function [9].

Data transmission efficiency will increase when more payload data is sent in one packet due to the package overhead which is 9 bytes per package in *VirtualWire* library. The efficiency of the data transmission can be defined as a ratio in percentage between the payload size and the total package size including package overhead (see Figure 5.8). We can make a conclusion that it will be better to send in one package as much useful data as possible and when we use encryption as closer to the block size. We can use this in practice for sensor node to collect data for a few periods of time and then transmit all the data in one package.

But this method will increase the latency of the system, so it will be not good for controlling applications like home automation, only for monitoring or data logging purposes.

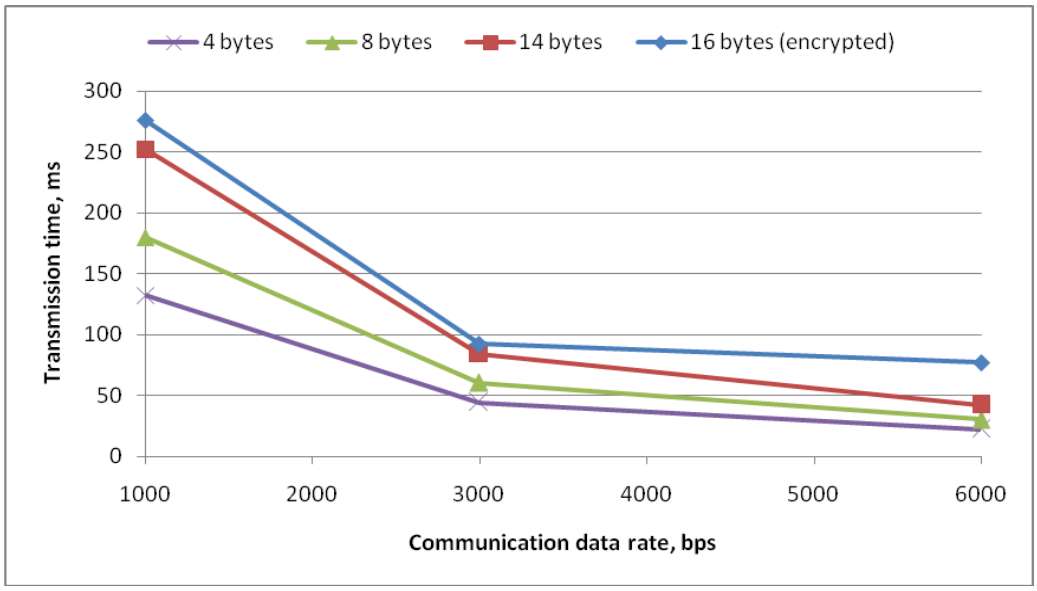



Figure 5.7: Comparison of transmission time for the plain text and encrypted message

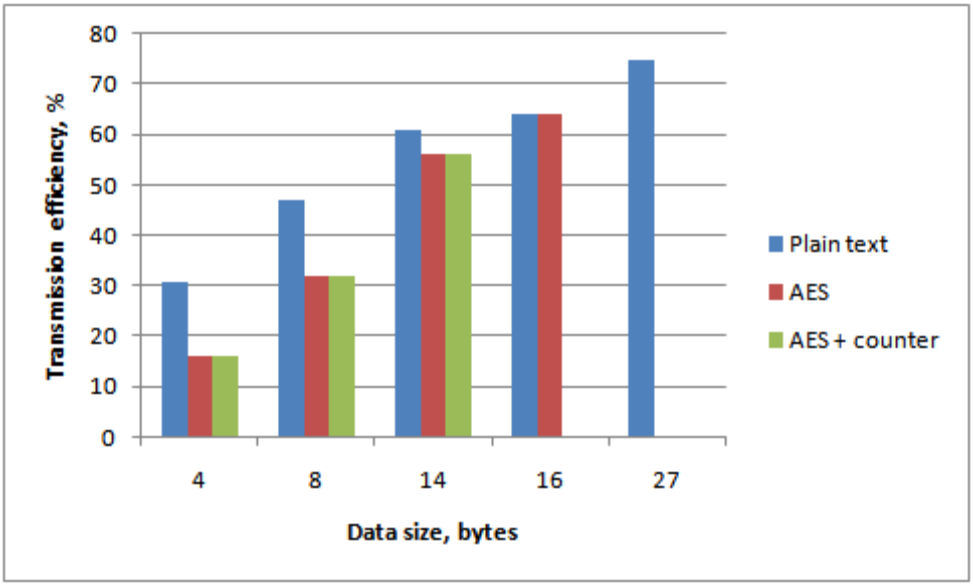



Figure 5.8: Transmission efficiency

**Power consumption**

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 $\mu$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 μA 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 5.2 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 | |
|---|---|---|---|---|---|
| *RF power, V* | *Data size, bytes* | - | + | - | + |
| 3.3 | 4 | 19 | 28 | 9.69 | 14.28 |
| | 16(enc) | 20 | 28 | 10.2 | 14.28 |
| 5 | 4 | 19.5 | 29.6 | 9.95 | 15.1 |
| | 16(enc) | 20 | 30 | 10.2 | 15.3 |

Table 5.1: Current consumption during the data transmission

| | | 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 5.2: Current consumption during the sleeping at different sleep modes

In Figure 5.9 you can see the comparison of measurement results using two different methods for different sleep modes.

During the experiment 3b we obtained the next results. First of all, it allowed us to visualize the current consumption of the device over time like in Figure 5.3. You can see from Figure 5.10 the comparison of the current consumption of Arduino Nano when it sends 4 bytes of sensor data without (5.10a) and with AES encryption (5.10b). It should be mentioned that almost all the time during the active stage device is sending data. For Figure 5.10a it is 44 ms and for 5.10b it's 92 ms. In the latter case, 16 bytes of encrypted data was sent.

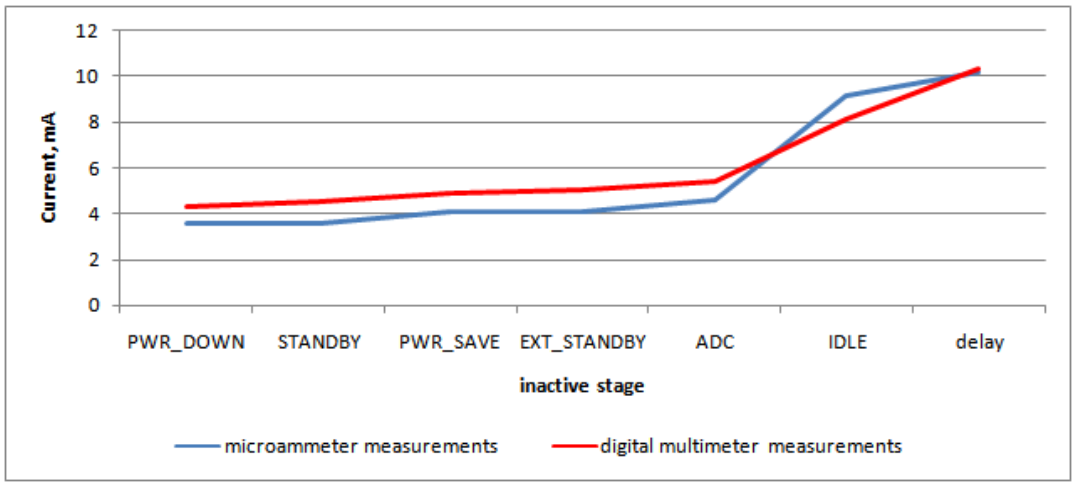
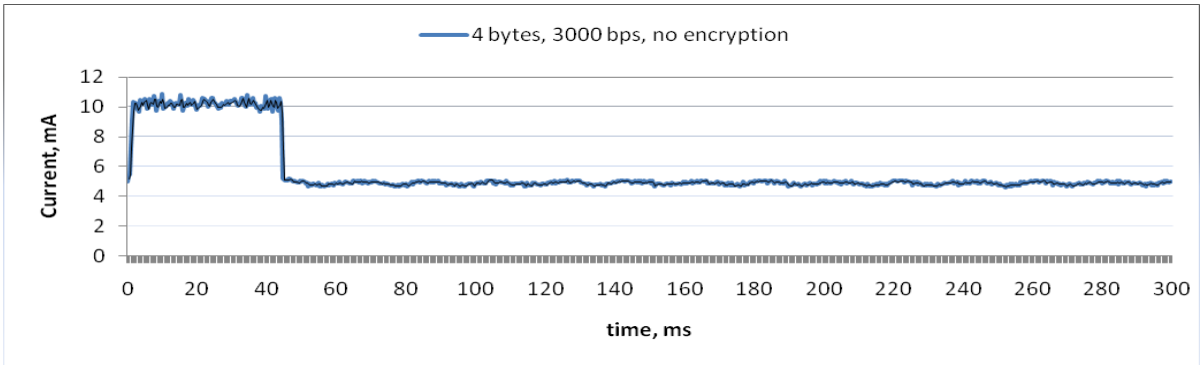
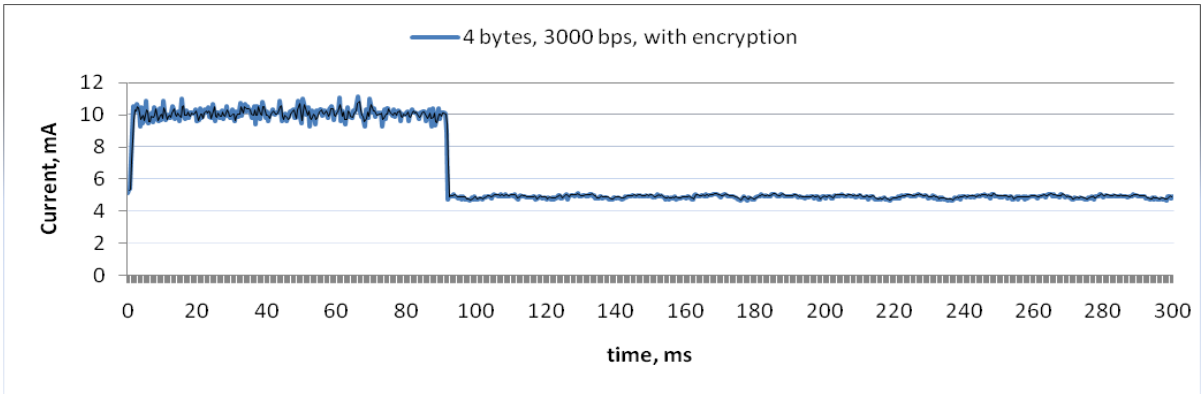



Figure 5.9: Comparison of measurement results of current for different sleep modes

As a voltage is significantly higher than a current in our scenario and doesn't change frequently, we will assume that it is a constant during the experiment. After measuring the voltage of the power supply with a multimeter we run a program for experiment 3b to calculate the energy consumption of the Arduino Nano in the scenario shown in Figure 5.10 using an equation 5.1. The results are shown in Figure 5.11.




(a) with no encryption




(b) with encryption

Figure 5.10: Current consumption of Arduino Nano microcontroller over time using sleep mode when sending 4 bytes of data in plain text (a) or using encryption (b)

So, when sending 16 bytes of data instead of 4 bytes of sensor data when using AES encryption it will reduce the lifetime of the battery. If 4 bytes of data should be sent every 300 ms the reduction will be 13.5 percent (about 1.2 mJ for one period). Because of the relatively short execution time of encryption in AES, the algorithm itself does not have so big influence on energy consumption.

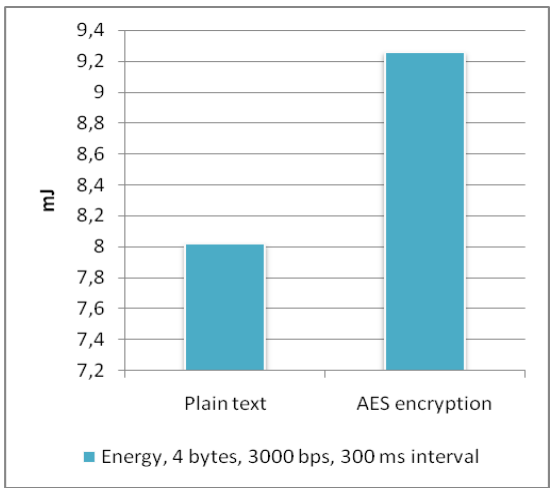



Figure 5.11: Comparison of consumed energy of Arduino Nano
when sending 4 bytes of data on data rate 3000 bps using sleep mode.
Considered time period equals 300 ms

But how long will the battery of the sensor node last?

The battery lifetime can be calculated using equation 5.2 based on a formula from [72].

$$Battery\ life\ (hours) = \frac{Battery\ capacity\ (mAh)}{\frac{I_{active}(mA) \cdot t_{active}(ms) + I_{sleep}(mA) \cdot t_{sleep}(ms)}{t_{active}(ms) + t_{sleep}(ms)}} \quad (5.2)$$

Let's assume we have a 5 V DC battery with capacity 1440 mAh. If we have a temperature sensor and we want to monitor the temperature it can be reasonable to send data from the sensor only once per five minutes. For controlling the air conditioner or a heater that period can vary. Heating devices, for example, are slow to respond the commands. Other measurements should be carried out continuously. A smart bulb which adjusts its brightness to the level of light inside the room (light sensor) or a motion (motion detector) can be an example. In Figures 5.12 and 5.13 you can see a summary of battery lifetime calculations depending on different parameters of communication based on values from Tables 5.1, 5.2 and Figure 5.7 using equation 5.2 for a period between each transmission of 5 minutes and 300 ms respectively.

So, 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 [26, 27] 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 [73]. 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.

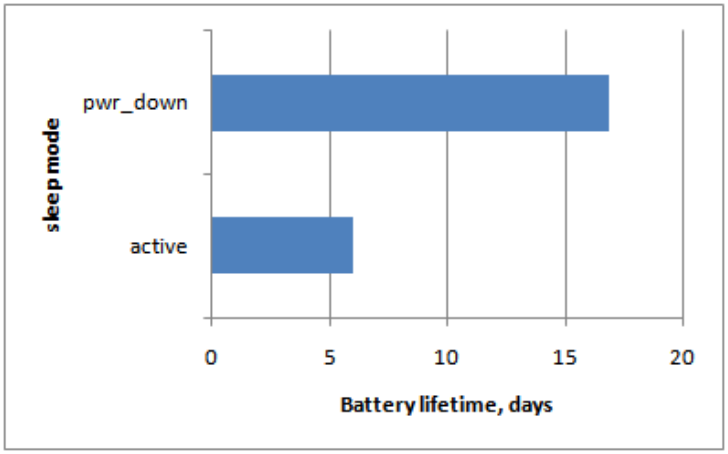



Figure 5.12: Battery lifetime for 5 minutes period

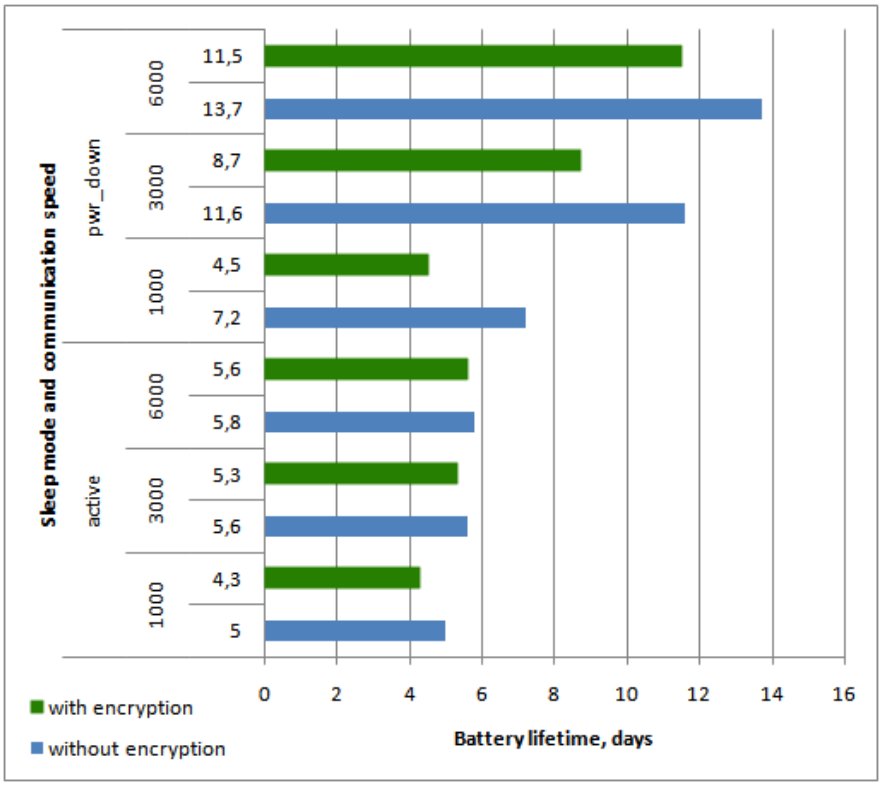



Figure 5.13: Battery lifetime for a period of 300 ms depending on encryption, sleep management, and communication speed

From Figure 5.13 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. In all cases using encryption reduces the lifetime of the battery. 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 5.1 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.

**Recommendations**

Based on the obtained results we can highlight following recommendations for the developers of home automation projects:

1. Advanced communication strategy (combination of techniques) for sensor node can be used:
   - average results are calculated for several periods and sent only once;
   - value is sent only when it is changed or exceed certain bounds;
   - send differential results and/or use data compression;
   - send few measurements (from one sensor or several different sensors) in one package to increase the efficiency of the communication. In that case, our security measures will not significantly reduce the productivity and lifetime of the sensor node;
   - change sleep period dynamically depending on the values of the sensor and the speed of their changes.
2. Symmetric encryption with smaller block size can be used (for example, 8 bytes in algorithms such as PRESENT, TEA, RC5, or HIGHT).
3. Use symmetric encryption with a counter in order to increase the level of security and protection from replay attacks.
4. Use maximum communication data rate that provides a reliable transmission between two nodes.
5. Use sleep management with PWR_DOWN sleep mode and other power saving techniques for Arduino.
6. Combine battery-operated sensor nodes and energy harvesting methods to get a self-powered and reliable device.
7. Adaptive cross-layer protocol stack can be developed which can adapt communication data rate, power of transmitter and receiver, and coding depending on channel characteristics and network conditions [49]. For example, if there are a lot of interferences in the channel, the application can migrate to the channel with fewer

interferences, lower data rate can be chosen (as a result we will get lower throughput) or transmitter power can be increased (as a result we will get better signal strength, increased range, higher throughput, and power consumption), different modulation and coding can be used to resist interferences in the channel.

# 6 CONCLUSION AND FUTURE WORK

In this research, we reviewed the current state of the Internet of Things, open issues, smart home, and requirements for home wireless communication. We also described and compared the main characteristics, performance, power consumption, and security of ZigBee, WiFi, Bluetooth, Z-Wave, and ANT communication technologies used in a smart home, gave recommendations on the use of wireless technologies for smart home applications.

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. So, it's important to investigate the other hardware and software strategies to minimize the power consumption of the microcontroller and maximize the lifetime of the device. More accurate measurements with better equipment should be done for different sleep modes.

Further investigation of advanced communication strategies can be a part of future work.

Further development of a home automation system based on DIY electronics (Arduino and Raspberry Pi) in full-scale size may be a part of a future work with a performance evaluation of the communication channel between the nodes using different wireless communication modules (RF 2.4 GHz, XBee, WiFi, Bluetooth) and different transmission parameters of communication (data rate, coding, supply voltage of a transmitter) while preserving the sufficient level of system performance.

# 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] V. Gunge and P. Yalagi, "Smart Home Automation: A Literature Review", *International Journal of Computer Applications*, pp. 6-10, 2016.

[5] M. Riley, *Programming your home*, 1st ed. Dallas, Tex: Pragmatic Bookshelf, 2012.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] O. Garcia-Morchon, R. Rietman, S. Sharma, L. Tolhuizen and J. Torre-Arce, "A Comprehensive and Lightweight Security Architecture to Secure the IoT Throughout the Lifecycle of a Device Based on HIMMO", in *Algorithms for Sensor Systems. Lecture Notes in Computer Science*, 1st ed., P. Bose, L. Gasieniec, K. Römer and R. Wattenhofer, Ed. Patras, Greece: 11th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS), 2015, pp. 112-128.

[11] O. Bello, S. Zeadally and M. Badra, "Network layer inter-operation of Device-to-Device communication technologies in Internet of Things (IoT)", *Ad Hoc Networks*, vol. 57, pp. 52-62, 2017.

[12] S. Raza, S. Duquennoy, J. Höglund, U. Roedig and T. Voigt, "Secure communication for the Internet of Things – a comparison of link-layer security and IPsec for 6LoWPAN", *Security and Communication Networks*, vol. 7, no. 12, pp. 2654-2668, 2012.

[13] N. Bressan, L. Bazzaco, N. Bui, P. Casari, L. Vangelista and M. Zorzi, "The Deployment of a Smart Monitoring System Using Wireless Sensor and Actuator

Networks", in *First IEEE International Conference on Smart Grid Communications (SmartGrid-Comm)*, 2010, pp. 49-54.

[14] 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.

[15] 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.

[16] M. Andersson, *Short-range Low Power Wireless Devices and Internet of Things (IoT)*, 1st ed. connectBlue, 2014, pp. 1-13.

[17] A. Rahman, "Comparison of Internet of Things (IoT) Data Link Protocols", pp. 1-21, 2015.

[18] F. Johari, "The security of communication protocols used for Internet of Things", Master of Science in Computer Science, Lund University, Sweden, 2015.

[19] C. Gomez and J. Paradells, "Wireless home automation networks: A survey of architectures and technologies", *IEEE Communications Magazine*, vol. 48, no. 6, pp. 92-101, 2010.

[20] T. Mendes, R. Godina, E. Rodrigues, J. Matias and J. Catalão, "Smart and energy-efficient home implementation: Wireless communication technologies role", in *5th International Conference on Power Engineering, Energy and Electrical Drives (POWERENG)*, 2015, pp. 377-382.

[21] P. Neelakanta and H. Dighe, "Robust factory wireless communications: a performance appraisal of the Bluetooth and the ZigBee colocated on an industrial floor", in *Industrial Electronics Society*, *2003. IECON '03. The 29th Annual Conference of the IEEE*, 2003, 2381-2386.

[22] C. Saad, B. Mostafa, E. Ahmadi and H. Abderrahmane, "Comparative Performance Analysis of Wireless Communication Protocols for Intelligent Sensors and Their Applications", *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 5, no. 4, pp. 76-85, 2014.

[23] M. Siekkinen, M. Hiienkari, J. Nurminen and J. Nieminen, "How Low Energy is Bluetooth Low Energy? Comparative Measurements with ZigBee/802.15.4", in *Wireless Communications and Networking Conference Workshops (WCNCW)*, Paris, France, 2012, pp. 232-237.

[24] J. Suhonen, "Designs for the Quality of Service Support in Low-Energy Wireless Sensor Network Protocols", Thesis for the degree of Doctor of Science in Technology, Tampere University of Technology, 2012.

[25] 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.

[26] 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].

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

[28] 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-embedded-microcontrollers. [Accessed: 02- May- 2017].

[29] H. Kwon, R. Rietman, J. Park and N. Kang, "Challenges in Deploying CoAP Over DTLS in Resource Constrained Environments", in *Information Security Applications. WISA 2015. Lecture Notes in Computer Science*, vol. 9503., H. Kim and D. Choi, Springer, Cham: 16th International Workshop on Information Security Applications(WISA), 2015, 269-280.

[30] A. Nordrum, "Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated", *IEEE Spectrum: Technology, Engineering, and Science News*, 2016. [Online]. Available: http://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated. [Accessed: 18- Mar- 2017].

[31] L. Columbus, "Internet Of Things Market To Reach $267B By 2020", *Forbes.com*, 2017. [Online]. Available: https://www.forbes.com/sites/louiscolumbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020. [Accessed: 19- Mar- 2017].

[32] "Expert Group on the Internet of Things (IoT-EG)", *European Commission*. [Online]. Available: http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=1752JeCyd173g&sig2=a3cHVzht3OtpsHdevmA87w. [Accessed: 17- Mar-2017].

[33] R. Hummen, J. Hiller, H. Wirtz, M. Henze, H. Shafagh and K. Wehrle, "6LoWPAN Fragmentation Attacks and Mitigation Mechanisms", in *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks (WiSec '13)*, Budapest, Hungary, 2013, pp. 55-66.

[34] C. Bormann, M. Ersue and A. Keranen, "RFC 7228 - Terminology for Constrained-Node Networks", *Tools.ietf.org*, 2014. [Online]. Available: https://tools.ietf.org/html/rfc7228. [Accessed: 27- Mar- 2017].

[35] "Chapter 3. Microprocessor Types and Specifications", *ptgmedia.pearsoncmg.com.* [Online]. Available: http://ptgmedia.pearsoncmg.com/images/9780789734044/samplechapter/0789734044_CH03.pdf. [Accessed: 15- Apr- 2017].

[36] M. Verle, *PIC Microcontrollers – Programming in Assembly*, 1st ed. MikroElektronika, 2008.

[37] F. Viani, F. Robol, A. Polo, P. Rocca, G. Oliveri and A. Massa, "Wireless Architectures for Heterogeneous Sensing in Smart Home Applications: Concepts and Real Implementation", *Proceedings of the IEEE*, vol. 101, no. 11, pp. 2381-2396, 2013.

[38] N. Komninos, E. Philippou and A. Pitsillides, "Survey in Smart Grid and Smart Home Security: Issues, Challenges and Countermeasures", *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1933-1954, 2014.

[39] V. Mohan, *An Introduction to Wireless M-Bus*, 1st ed. Silicon Labs, 2017, pp. 1-19.

[40] D. Han and J. Lim, "Smart home energy management system using IEEE 802.15.4 and zigbee", *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1403-1410, 2010.

[41] J. Dodge, "What is a smart appliance, anyway? | ZDNet", *ZDNet*, 2009. [Online]. Available: http://www.zdnet.com/article/what-is-a-smart-appliance-anyway/. [Accessed: 19- Mar- 2017].

[42] "What is the Smart Grid?", *Smartgrid.gov*. [Online]. Available: https://www.smartgrid.gov/the_smar_grid/. [Accessed: 03- Mar- 2017].

[43] "11 Internet of Things (IoT) Protocols You Need to Know About", *Rs-online.com*, 2015. [Online]. Available: https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about. [Accessed: 05- Feb- 2017].

[44] "IoT Technology | 2017 Overview Guide on Protocols, Software, Hardware and Network Trends", *Postscapes.com*, 2017. [Online]. Available: https://www.postscapes.com/internet-of-things-technologies/. [Accessed: 01-Apr-2017].

[45] M. Mahmoud and A. Mohamad, "A Study of Efficient Power Consumption Wireless Communication Techniques/Modules for Internet of Things (IoT) Applications", *Advances in Internet of Things*, vol. 6, no. 2, pp. 19-29, 2016.

[46] E. Ferro and F. Potorti, "Bluetooth and wi-fi wireless protocols: a survey and a comparison", *IEEE Wireless Communications*, vol. 12, no. 1, pp. 12-26, 2005.

[47] "Five Fundamentals of RF You Must Know for WLAN Success", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=dwDRAqfA7GI. [Accessed: 02-Apr-2017].

[48] "Introduction to Wi-Fi and RF", *YouTube*, 2014. [Online]. Available: https://www.youtube.com/watch?v=CthZCSmQGmw. [Accessed: 02- Apr- 2017].

[49] A. Goldsmith, *Wireless communication*, 1st ed. California: Cambridge University Press, 2005, pp. 1-471.

[50] "Smart Homes | zigbee alliance", *Zigbee.org*. [Online]. Available: http://www.zigbee.org/what-is-zigbee/494-2/. [Accessed: 07- Apr- 2017].

[51] M. Franceschinis, C. Pastrone, M. Spirito and C. Borean, "On the performance of ZigBee Pro and ZigBee IP in IEEE 802.15.4 networks", in *9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Lyon, 2013, pp. 83-88.

[52] T. Aasebø, "Wireless technologies: Bluetooth, ZigBee and ANT".

[53] "About Z-Wave Technology - Z-Wave Alliance", *Z-Wave Alliance*. [Online]. Available: http://z-wavealliance.org/about_z-wave_technology/. [Accessed: 03- Apr- 2017].

[54] *Wireless connectivity for IoT applications*, 1st ed. ST, 2017, pp. 1-48.

[55] I. Glaropoulos, V. Vukadinovic and S. Mangold, "Contiki80211: An IEEE 802.11 Radio Link Layer for the Contiki OS", in *High Performance Computing and Communications, 6th Intl Symp on Cyberspace Safety and Security*, *11th Intl Conf on Embedded Software and Syst (HPCC,CSS,ICESS)*, Paris, 2014, pp. 621-624.

[56] "Wi-Fi Alliance", *Wi-fi.org.* [Online]. Available: http://www.wi-fi.org/. [Accessed: 24- Apr- 2017].

[57] J. Penttinen, *Wireless communications security*, 1st ed. John Wiley & Sons, 2016, pp. 1-336.

[58] N. Mehmood and R. Culmone, "An ANT+ Protocol Based Health Care System", in *29th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, Gwangiu, 2015, pp. 193-198.

[59] IEEE Standard for Local and metropolitan area networks - Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), 1st ed. New York, USA: The Institute of Electrical and Electronics Engineers, Inc., 2011, pp. 1-314.

[60] "zigbee PRO with Green Power | zigbee alliance", *Zigbee.org*. [Online]. Available: http://www.zigbee.org/zigbee-for-developers/network-specifications/zigbeepro/. [Accessed: 08- Apr- 2017].

[61] "zigbee Home Automation | zigbee alliance", *Zigbee.org.* [Online]. Available: http://www.zigbee.org/zigbee-for-developers/applicationstandards/zigbeehomeautomation/. [Accessed: 07- Apr- 2017].

[62] "zigbee Smart Energy | zigbee alliance", *Zigbee.org.* [Online]. Available: http://www.zigbee.org/zigbee-for-developers/applicationstandards/zigbeesmartenergy/. [Accessed: 08- Apr- 2017].

[63] "zigbee IP and 920IP | zigbee alliance", *Zigbee.org.* [Online]. Available: http://www.zigbee.org/zigbee-for-developers/network-specifications/zigbeeip/. [Accessed: 10- Apr- 2017].

[64] D. Gascón, "Security in 802.15.4 and ZigBee networks | Libelium", *Libelium.com*, 2009. [Online]. Available: http://www.libelium.com/security-802-15-4-zigbee/. [Accessed: 14- Apr- 2017].

[65] G. Dini and M. Tiloca, "Considerations on Security in ZigBee Networks", in *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC)*, 2010, pp. 58-65.

[66] "Arduino - Home", *Arduino.cc.* [Online]. Available: https://www.arduino.cc/. [Accessed: 09- Apr- 2017].

[67] P. Traeg, "IoT Projects: Raspberry Pi vs Arduino", *Universal Mind*, 2015. [Online]. Available: http://www.universalmind.com/blog/raspberry-pi-vs-arduino-when-to-use-which/. [Accessed: 19- Apr- 2017].

[68] L. Orsini, "Arduino Vs. Raspberry Pi: Which Is The Right DIY Platform For You?", *ReadWrite*, 2014. [Online]. Available: http://readwrite.com/2014/05/07/arduino-vs-raspberry-pi-projects-diy-platform/. [Accessed: 20- Apr- 2017].

[69] "Wireless transmitter + receiver 433Mhz", *Iarduino.ru.* [Online]. Available: http://iarduino.ru/shop/Expansion-payments/besprovodnoy-peredatchik-priemnik-315-433mhz-dc5v.html. [Accessed: 01- May- 2017].

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

[71] D. Landman, "Arduino AESLib", *GitHub*, 2016. [Online]. Available: https://github.com/DavyLandman/AESLib. [Accessed: 01- May- 2017].

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

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