

UNIVERSITY OF VAASA

SCHOOL OF TECHNOLOGY AND INNOVATION

WIRELESS INDUSTRIAL AUTOMATION

Akpojoto Akporido Siemuri

**ROBUST AND RELIABLE WIRELESS COMMUNICATION BETWEEN
SMART NOX SENSOR AND THE SPEEDGOAT/ENGINE CONTROL
MODULE**

A case study of Wärtsilä's smart NOx sensor and W4L20 Diesel Engine

Master`s thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, 13 February, 2019.

Supervisor
Instructors

Professor Timo Mantere
M.Sc. Tobias Glocker
Assistant Professor Mike Mekkanen

VAASA 2019

ACKNOWLEDGEMENT

First, I am thankful to God for His wisdom and guidance through my master's studies. I would like to thank Professor Mohammed Elmusrati and Reino Virrankoski for the opportunity to work on the Wärtsilä smart NO_x sensor case under the Work Packet 3 (WP3 – Wireless Communication) in the Smart Energy Systems Research Platform (SESP) project. I am also thankful to Professor Kimmo Kauhaniemi and Assistant Professor Mike Mekkanen for an excellent guidance through the SESP WP3 project to its completion.

I will like to mention my gratitude to Professor Timo Mantere and Tobias Glocker for an excellent supervision and guidance in achieving success in the completion of my thesis. I am also thankful to the laboratory engineers in Technobothnia, Veli-Matti Eskonen and Juha Miettinen, for providing all the necessary equipment and a good working environment and to Xiaoguo Storm for helping with the smart NO_x/speedgoat tests in VEBIC. Thanks to Rayko Toshev, Sulaymon Tajudeen and Ibukun Odubogun for giving access to the digital manufacturing laboratory in Technobothnia and providing the relevant equipment to achieve the 3D printing aspects applied in this thesis and to Sulaymon Tajudeen for assisting in the design of the models that was 3D printed.

Lastly, I am thankful to everyone who supported me in any way towards the completion of this thesis.

TABLE OF CONTENTS

	page
ACKNOWLEDGEMENT	1
LIST OF FIGURES	5
LIST OF TABLES	8
ABBREVIATIONS	10
ABSTRACT	13
1. INTRODUCTION	15
1.1. Motivation	15
1.2. Objectives	16
1.3. Methods	16
1.4. Thesis Structure	17
2. INTRODUCTION OF PROTOCOLS AND MAJOR COMPONENTS	18
2.1. Controller Area Network (CAN)	18
2.1.1. The CAN Bus	19
2.1.2. CAN Standard	20
2.1.3. CAN Messages	22
2.2. Wireless Communication Protocols	24
2.2.1. Factors that affect wireless communication	25
2.2.2. Types of Wireless Communication Protocols	27
2.2.3. Bluetooth Low Energy (BLE)	28
2.2.4. Zigbee (IEEE 802.15.4)	32
2.2.5. WiFi (IEEE 802.11 b/g/a)	37
2.2.6. LoRa (Long Range)	40
2.2.7. Comparing the Wireless Communication Protocols	47
2.2.8. Choosing a Wireless Protocol	49
2.2.9. Basic Network Attacks	50

2.2.10. Encryption and Authentication	52
2.3. Smart NOx Sensor, Speedgoat and Engine Control Module (ECM)	52
2.3.1. Smart NOx Sensor	52
2.3.2. Acquiring data from the Smart NOx sensor	55
2.3.3. Testing the Smart NOx sensor	56
2.3.4. Calculating O ₂ % and NOx ppm	57
2.3.5. Speedgoat and the Engine Control Module (ECM)	57
3. SMART NOX AND SPEEDGOAT WIRELESS COMMUNICATION	58
3.1. System Architecture	59
3.2. System Overview	73
3.2.1. Connecting the Smart NOx Sensor	73
3.2.2. The BLE-CAN bridge Hardware	73
3.2.3. The BLE-CAN bridge Software	76
3.2.4. The XBee-CAN bridge Hardware	78
3.2.5. The XBee-CAN bridge Software	80
3.2.6. The WIFI-CAN bridge Hardware	83
3.2.7. The WIFI-CAN bridge Software	85
3.2.8. The LoRa-CAN bridge Hardware	87
3.2.9. The LoRa-CAN bridge Software	89
3.2.10. Flowchart for codes and Viewing the CAN frames	91
3.2.11. Connecting to the Speedgoat	95
3.3. Wireless Communication Performance Measure	96
4. EXPERIMENT AND ANALYSIS	97
4.1. Details of Transmitted Payload and LCD Display for XBee-CAN Modules	98
4.2. Bluetooth Low Energy (BLE)	99
4.2.1. BLE RSSI Values	99
4.2.2. BLE Packet Loss	100
4.2.3. BLE Latency	101
4.3. XBee (IEEE 802.15.4)	101
4.3.1. XBee RSSI Values	101
4.3.2. XBee Packet Loss	102

4.3.3. XBee Latency	103
4.4. WIFI (IEEE 802.11 b/g/a)	103
4.4.1. WIFI RSSI Values	103
4.4.2. WIFI Packet Loss	104
4.4.3. WIFI Latency	105
4.5. LoRa (Long Range)	105
4.5.1. LoRa RSSI Values	105
4.5.2. LoRa Packet Loss	106
4.5.3. LoRa Latency	107
4.6. Bit Error Check for all wireless protocols	107
4.7. Security Implementation	108
4.8. Power Consumption	108
4.9. Comparing the Wireless Solutions Based on the Analysis of Results	109
4.10. Viewing O ₂ % and NO _x ppm Values on the Speedgoat	114
4.11. SmartNO _x + XBee-CAN Module Test on Wärtsilä W4L20 Diesel Engine	115
4.12. Applying Additive Manufacturing (3D printing) to the Designed Prototype	120
5. CONCLUSION AND FUTURE WORK	121
LIST OF REFERENCES	124
APPENDICES	134
APPENDIX 1. Schematic of Mikroelektronika CAN SPI click board	134
APPENDIX 2. Smart NO _x , XBee-CAN Module and Speedgoat system overview	135
APPENDIX 3. LCD Display for Transmitter/Receiver Modules	136
APPENDIX 4. Sample output of transmitter and receiver code	137
APPENDIX 5. 3D printed protective casing body	138
APPENDIX 6. 3D printed protective casing covers	138

LIST OF FIGURES

Figure 1. CAN Bus Architecture	19
Figure 2. ISO11898 Architecture	20
Figure 3. CAN 2.0A - Standard CAN Frame 11-Bit Identifier	21
Figure 4. CAN 2.0B - Extended CAN Frame 29-Bit Identifier	22
Figure 5. Arbitration on a CAN Bus	23
Figure 6. Fresnel zone illustration	25
Figure 7. Bluetooth Low Energy Frequency Channels	30
Figure 8. Zigbee Packet Structure	33
Figure 9. ZigBee Protocol Stack Architecture	34
Figure 10. Application Layer Security	36
Figure 11. Network Layer Security	36
Figure 12. Application and Network Layer Security	37
Figure 13: LoRa Packet Structure	42
Figure 14. A Simplified SX1272 Block Diagram	45
Figure 15. LoRa Network Architecture	46
Figure 16. Types of network attacks	51
Figure 17. Schematic representation of an amperometric NO _x sensor	54
Figure 18. CAN frame to start heating smart NO _x	56
Figure 19. Engine Control Unit of a 1996 Chevrolet Beretta	58
Figure 20. CAN Bus Module	60
Figure 21. Multiprotocol Radio Shield v2.0	61
Figure 22. Arduino UNO Rev.3	62
Figure 23. Arduino IDE	63

Figure 24. Wasmote development board	63
Figure 25. Wasmote IDE	64
Figure 26. Wasmote Expansion Board	64
Figure 27. XBee PRO Module	66
Figure 28. XBee Explorer USB	66
Figure 29. XBee PRO Module on XBee Explorer USB	66
Figure 30. XCTU tool	67
Figure 31. LoRa Module	68
Figure 32. WIFI PRO Module	69
Figure 33. Wasmote Bluetooth Low Energy module	69
Figure 34. Smart NOx sensor from Wärtsilä	71
Figure 35. Speedgoat – A performance real-time target machine	71
Figure 36. Simulink model to receive smart NOx CAN frames	72
Figure 37. Hardware setup of BLE-CAN bridge	74
Figure 38. Block diagram for the hardware setup of BLE-CAN bridge	75
Figure 39. Smart NOx sensor and BLE-CAN transmitter	77
Figure 40. BLE-CAN receiver and Kvaser Leaf Light HS v2 USB	77
Figure 41. Hardware setup of Xbee-CAN bridge	78
Figure 42. Block diagram for the hardware setup of XBee-CAN bridge	79
Figure 43. Smart NOx sensor and XBee-CAN transmitter	80
Figure 44. XBee-CAN receiver and Kvaser Leaf Light HS v2 USB	81
Figure 45. Hardware setup of WIFI-CAN bridge	83
Figure 46. Block diagram for the hardware setup of WIFI-CAN bridge	84
Figure 47. Smart NOx sensor and WIFI-CAN transmitter	85

Figure 48. WIFI-CAN receiver and Kvaser Leaf Light HS v2 USB	86
Figure 49. Hardware setup of LoRa-CAN bridge	87
Figure 50. Block diagram for the hardware setup of LoRa-CAN bridge	88
Figure 51. Smart NOx sensor and LoRa-CAN transmitter	89
Figure 52. LoRa-CAN receiver and Kvaser Leaf Light HS v2 USB	90
Figure 53. Flowchart for the XBee, WiFi and LoRa transmitter codes	92
Figure 54. Flowchart for the XBee, WiFi and LoRa receiver codes	93
Figure 55. Hexadecimal View of the CAN frames	94
Figure 56. Decimal View of the CAN frames	94
Figure 57. Continuously updated sliding graph for O2% and NOx ppm values	95
Figure 58. Transmitted Smart NOx Payload	98
Figure 59. RSSI Measurements for all the wireless protocol in Technobothnia	111
Figure 60. RSSI Measurements for all the wireless protocol in VEBIC	112
Figure 61. Continuously updated sliding graph for O2% and NOx ppm values	115
Figure 62. Speedgoat result when Wärtsilä W4L20 Diesel Engine is idle	117
Figure 63. Wärtsilä W4L20 Diesel Engine is running without load	117
Figure 64. Speedgoat sliding graph of the O2% and NOx ppm values	118
Figure 65. Speedgoat results for O2% and NOx ppm values	118
Figure 66. Wärtsilä W4L20 Diesel Engine is running with load	119
Figure 67. Speedgoat sliding graph of the O2% and NOx ppm values	119
Figure 68. Speedgoat results for O2% and NOx ppm values	119
Figure 69. XBee-CAN Receiver/Transmitter in 3D printed protective casings	121

LIST OF TABLES

Table 1. Main Features of BLE that differ from standard Bluetooth	29
Table 2. BLE Radio feature	31
Table 3. LoRa Device Variants and Key Parameters taken from LoRa SX1272/73 Datasheet, Rev. 3.1. Semtech, 2017	42
Table 4. Comparing BLE, XBee, WIFI and LoRa Wireless Protocols	48
Table 5. Payload in smart NOx CAN frames	56
Table 6. Technical details of the CAN Bus Module	60
Table 7. XBee 802.15.4 Channel Number Frequency	65
Table 8. LoRa specification	68
Table 9. Main features of the BLE module	70
Table 10. NOx sensor performance specification	70
Table 11. Smart NOx sensor pin labeling	73
Table 12. BLE maximum and minimum RSSI measurement in Technobothnia	99
Table 13. BLE maximum and minimum RSSI measurement in VEBIC	99
Table 14. BLE maximum and minimum Packet Loss measurement in Technobothnia	100
Table 15. BLE maximum and minimum Packet Loss measurement in VEBIC	100
Table 16. BLE maximum and minimum latency measurement in milliseconds	101
Table 17. XBee maximum and minimum RSSI measurement in Technobothnia	102
Table 18. XBee maximum and minimum RSSI measurement in VEBIC	102
Table 19. XBee maximum and minimum Packet Loss measurement in Technobothnia	102
Table 20. XBee maximum and minimum Packet Loss measurement in VEBIC	102

Table 21. XBee maximum and minimum latency measurement in milliseconds	103
Table 22. WIFI maximum and minimum RSSI measurement in Technobothnia	104
Table 23. WIFI maximum and minimum RSSI measurement in VEBIC	104
Table 24. WIFI maximum and minimum Packet Loss measurement in Technobothnia	104
Table 25. WIFI maximum and minimum Packet Loss measurement in VEBIC	105
Table 26. WIFI maximum and minimum latency measurement in milliseconds	105
Table 27. LoRa maximum and minimum RSSI measurement in Technobothnia	106
Table 28. LoRa maximum and minimum RSSI measurement in VEBIC	106
Table 29. LoRa maximum and minimum Packet Loss measurement in Technobothnia	106
Table 30. LoRa maximum and minimum Packet Loss measurement in VEBIC	107
Table 31. LoRa maximum and minimum latency measurement in milliseconds	107
Table 32. Computed Battery Life of Transmitter and Receiver Modules	109
Table 33. Comparison of the wireless protocols based on the analysis of results	110
Table 34. Comparison of the values from SICK and Smart NOx sensor for the Wärtsilä W4L20 Diesel Engine for different operation modes	116

ABBREVIATIONS

AES	Advanced Encryption Standard
BLE	Bluetooth Low Energy
BER	Bit Error Rate
CAD	Computer Aided Design
CAN	Controller Area Network
CBC	Cipher Block Chaining
dBm	decibel-milliwatts
DDM	Direct Digital Manufacturing
ECM	Engine Control Module
ECU	Engine Control Unit
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Standard Organization
LCD	Liquid Crystal Display
LoRa	Long Range
mAh	Milliampere hour
MCP2515	Microchip
ms	Milliseconds
NO _x	Nitrogen oxide
O ₂	Oxygen
OSI	Open System Interconnection
OTA	Over the Air
ppm	parts per million
QoS	Quality of Service
RSSI	Received Signal Strength Indicator
SAE	Society of Automotive Engineers
SPI	Serial Peripheral Interface
TCP/IP	Traffic Control Protocol/Internet Protocol
μA	Microampere
UART	Universal Asynchronous Receiver/Transmitter

USB	Universal Serial Bus
VEBIC	Vaasa Energy Business Innovation Centre
WEP	Wired Equivalent Privacy
WiFi	Wireless Fidelity
WSN	Wireless Sensor Network
ZC	ZigBee Coordinator
ZDO	ZigBee Device Object
ZED	ZigBee End Device
ZR	ZigBee Router

UNIVERSITY OF VAASA
School of Technology and Innovation

Author:	Akpojoto Siemuri
Topic of the Thesis:	Robust and Reliable Wireless Communication Between the Smart NO _x Sensor and the Speedgoat/Engine Control Module
Supervisor:	Professor Timo Mantere
Instructors:	M.Sc. Tobias Glocker Assistant Professor Mike Mekkanen
Department:	Department of Computer Science
Degree:	Master of Science in Technology
Degree Programme:	Wireless Industrial Automation
Year of entering the University:	2016
Year of completing the thesis:	2019
Number of pages:	138

ABSTRACT

In recent years, the industrial applications of the wireless transmission of data acquired through sensors have been growing. Addressing the challenges or requirements that come with this needs the integration of new product designs and manufacturing techniques with automation devices. Factors like development time, security, reliability, transmission in an industrial environment, data rate, battery life with energy harvesting capabilities, etc. are of major concerns.

This thesis is based on the Wärtsilä smart NO_x sensor case study which investigates the possibility of replacing the existing wired CAN bus connection between the smart NO_x sensor and the rapid control prototyping system speedgoat and possibly in the future the Engine Control Unit (ECU) with a wireless communication solution. The designed prototype would wirelessly transmit the smart NO_x sensor data. The smart NO_x sensor data is received using a CAN bus integrated with a wireless transmitter module. The wireless receiver module receives the data and then relays the CAN frames through an integrated CAN Bus to the speedgoat. A matlab simulink module has been programmed into the speedgoat to receive the CAN frames, calculate O₂% and NO_x ppm values and display the results on a monitor connected to the speedgoat. Criteria like transmission in industrial environments, packet loss, RSSI, bit error rate, reliability and security of the wireless solution are analyzed. According to the analysis done and best practices, a wireless solution is recommended and implemented. The wireless-CAN prototype is installed on the Wärtsilä W4L20 diesel engine in VEBIC for monitoring and observation.

KEY WORDS: BLE, CAN Bus, Engine Control Module (ECM), LoRa, RSSI: Received Signal Strength Indicator, Smart NO_x sensor, Speedgoat, Wi-Fi, Wireless Communication, ZigBee

1. INTRODUCTION

Modern industries's rapid development and increase in the economics of scale leads to production and industrial automation. These brings about the need to transfer data and the integration of data. This can be achieved using wireless communication, therefore, analysis of some well-known wireless communication solutions is crucial in achieving reliable and flexible data transfer. (Gao, Huang, Chen, Jin, & Luo 2013.)

Wireless connectivity offers multiple advantages such as easy installation and maintenance, better flexibility and scalability and long communication range. However, wireless communication introduces new challenges and risks such as noise and interference which might cause transmission errors, delays or connection drops. It is also prone to malicious attackers that might attempt to spy, hack into to controls or interfere with and jam communications. Therefore, careful considerations and field testing is required to verify if a wireless solution can deliver the expected robustness and security compared to the wired solution.

1.1. Motivation

The approach taken in this thesis is based on the case study of Wärtsilä's smart NOx sensor. They are interested in limiting hard wire cabling and possibly moving to wireless communication between the sensors and the speedgoat or Engine Control Module (ECM). In our case study, the smart NOx sensor is connected to the engine control unit (ECU) with a wired CAN bus connection. Data is transmitted using SAE J1939 protocol which is built on top of CAN Networks. SAE J1939 is developed specifically for use in heavy duty environments, with an emphasize on achieving reliable and fault tolerant communications.

1.2. Objectives

This thesis investigates the possibility of replacing the existing wired CAN bus connection between the smart NO_x sensor and the rapid control prototyping system speedgoat and possibly in the future the Engine Control Unit (ECU) with a wireless communication solution. For the purpose of comparison, some wireless protocols are implemented and analyzed with the aim of coming up with recommended wireless solutions. These recommendations must achieve and agree with some criteria like transmission in industrial environments, packet loss rate, RSSI, bit error rate, reliability and security of the wireless solution etc.

Guidelines: The designed prototype should wirelessly transmit the smart NO_x sensor data. The smart NO_x sensor data is received using a CAN bus integrated to a wireless transmitter module. The wireless receiver module receives the data and then relays the CAN frames through integrated CAN Bus to the speedgoat. A matlab simulink module has been programmed into the speedgoat to receive CAN frames, calculate O₂% and NO_x ppm and display the results on a monitor connected to the speedgoat.

Specifications: According to the prototype design plan, the following are to be considered; development time, overall cost, transmission in industrial environment, transmission rate, battery life with energy harvesting capabilities and low energy consumption, lifetime of the technology, future prospect of the technology, backwards compatibility of the technology and the feasibility of implementing the solution as a final product.

1.3. Methods

The smart NO_x is connected to the CAN bus at the transmitter side. The CAN bus is interfaced with the wireless device (BLE, ZigBee, WiFi and LoRa) over an expansion board or multi-protocol radio shield which allows for connection of two communication modules at the same time. The hardware setup is programmed to read the data coming from the smart NO_x sensor through the CAN bus and transfer the data through SPI to the wireless module for wireless transmission to the receiver side. At the receiver side,

the device is programmed to transfer the data received by the wireless module to the CAN bus and then the data is sent from the CAN bus to the speedgoat which is connected to it. Each wireless solution is implemented separately in turns and analyzed. To achieve this, measurements such as Receiver Signal Strength Indicator (RSSI), packet delivery rate, bit error rate, and latency were taken and used for comparing the implemented wireless protocols.

1.4. Thesis Structure

This thesis has five chapters. Chapter 1 introduces the research topic presenting the objective and motivation of this thesis as well as the methods used.

Chapter 2 presents the theoretical review of how the smart NO_x sensor and the speedgoat works. It also presents the Control Area Network (CAN) protocol with details about the CAN standard and its features and the selected wireless communication protocols. The wireless communication protocols used in this thesis includes BLE, LoRa, WiFi and ZigBee. A comparison of the wireless communication protocols in terms of frequency, range, maximum data rate, power sources options and most appropriate uses of the wireless solution is done.

Chapter 3 presents the description of the thesis topic and how the wireless communication between the smart NO_x sensor and the speedgoat can be achieved for each of the wireless solution. Chapter 4 describes the interfacing of the smart NO_x sensor and the speedgoat to each wireless module using an external CAN bus. It also presents simulation and analysis of the results obtained for each wireless protocol as well as specific measurements such as latency, delivery rate and bit error rate, etc. The research conclusion, recommendations and possible future study based on the results in chapter 4 are presented in chapter 5.

The appendix contains the pictures of the hardware implementations done as well as extracts from the codes used in programming the transmitter and receiver wireless CAN communication modules.

2. INTRODUCTION OF PROTOCOLS AND MAJOR COMPONENTS

This chapter presents the theoretical background of the communication protocols used as well as the major components and principles applied in this thesis. Section 2.1 introduces the control area network, section 2.2 introduces the wireless protocols used in the thesis work and section 2.3 briefly presents other components like the smart NO_x sensor, speedgoat and Engine Control Module (ECM).

2.1. Controller Area Network (CAN)

Unlike USB or Ethernet that sends large blocks of data point-to-point from a node A to node B with supervision from a central bus master, CAN network broadcast several short messages like temperature reading, or RPM to the entire network. This provides data consistency in every node of the system. The controller area network (CAN) is suitable for the various high-level industrial protocols embracing CAN and the ISO 11898:2003 standard as their physical layer. It has tremendous flexibility in system design due to its cost, performance, and upgradeability. (Texas Instrument 2016.)

CAN is a solution for automation industries and the CAN protocol is used in systems that need to transmit and receive a small amount of data with real-time requirements. CAN protocol has been stipulated as an international standard by 150 International Standard Organizations. (Wan, Xing & Cai 2009.)

CAN transmits signals on the CAN network using two wires, CAN-High and CAN-Low. These 2 wires operate in different mode carrying inverted voltages which decrease noise interference. The standard being used determines the voltage level and other characteristics of the physical layer. The two standards are the ISO11898 (CAN High Speed) standard and the ISO11519 (CAN Low Speed) standard. (Nilsson 2018.)

The international standard ISO11898 definition of CAN bus state that, it is a fully digital field control devices connection bus, which can efficiently support the serial com-

munication of distributed control and real-time systems. CAN bus is widely used with sensors for data acquisition, industrial control systems and is an instrument with high reliability and flexibility. (Texas Instrument 2016.)

2.1.1. The CAN Bus

Robert Bosch developed the automotive CAN Bus. It is a multi-master message broadcast system that gives a maximum signaling rate of 1 Megabit per second (Mbps). Automotive components use it to communicate on a single or dual-wire networked data bus. CAN is a serial bus protocol used to connect individual systems and sensors and it is an alternative to conventional multi-wire looms. (Texas Instrument 2016.)

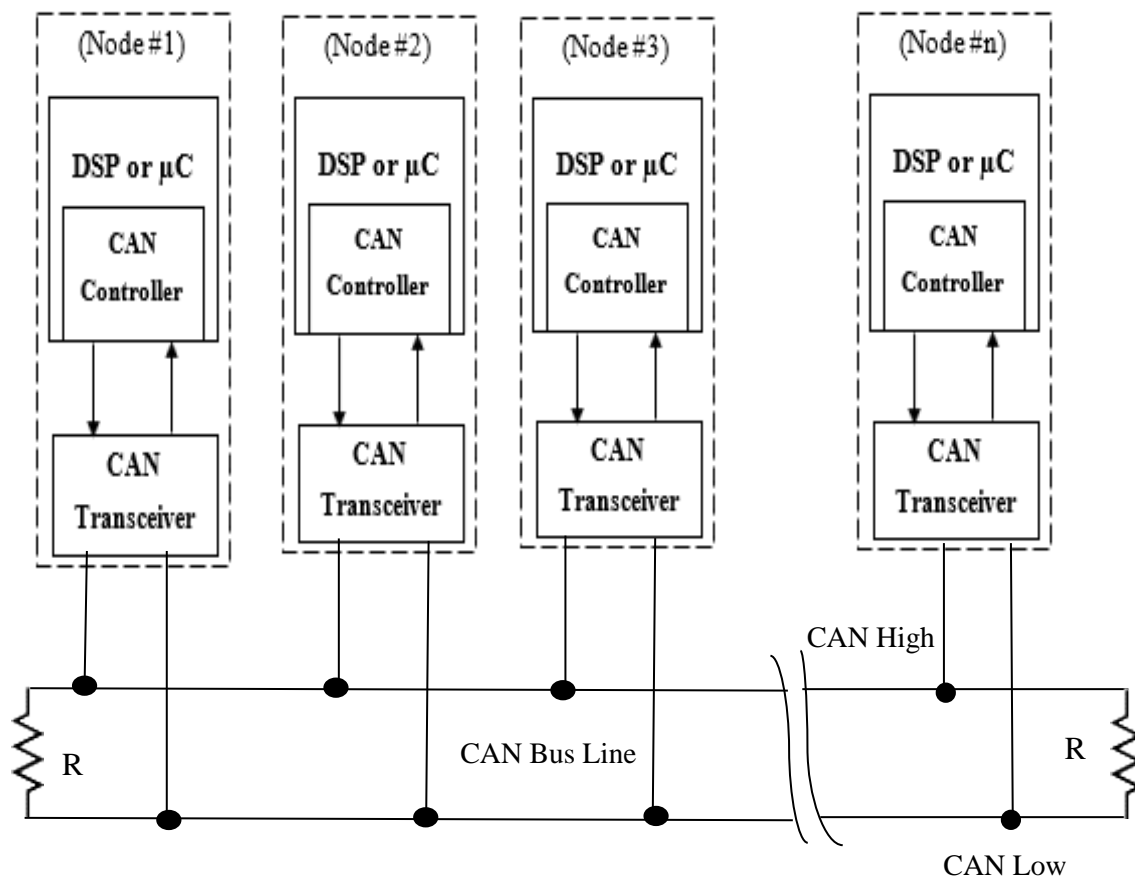


Figure 1. CAN Bus Architecture (Github 2018).

Figure 1 shows the CAN Bus Architecture. The maximum signaling rate of 1Mbps is achieved with the High-Speed ISO11898 standard specifications having a bus length of 40m and maximum of 30 nodes. The cable could be a shielded or unshielded twisted-pair having a 120- Ω resistor at each end. This standard uses a single line of twisted-pair cable as the network topology as presented in figure 1. A 120- Ω resistors is used to terminate both ends matching the characteristic impedance of the line to prevent signal reflections. Using RL on a node should be avoided based on the ISO 11898 because the node will be disconnected from the bus and the bus lines would lose termination. (Texas Instrument 2016.)

2.1.2. CAN Standard

The ISO 11898:2003 CAN communication protocol gives details on how information is transmitted from one device to another on a network and comply with the Open System Interconnect (OSI) model. The Open System Interconnect (OSI) model is defined in terms of layer where the physical layer of the module defines the actual communication between devices connected by the physical medium. The last two layers of the OSI/ISO model's seven layers are defined by the ISO 11898 architecture as the data-link layer and the physical layers respectively as shown in figure 2. (Texas Instrument 2016.)

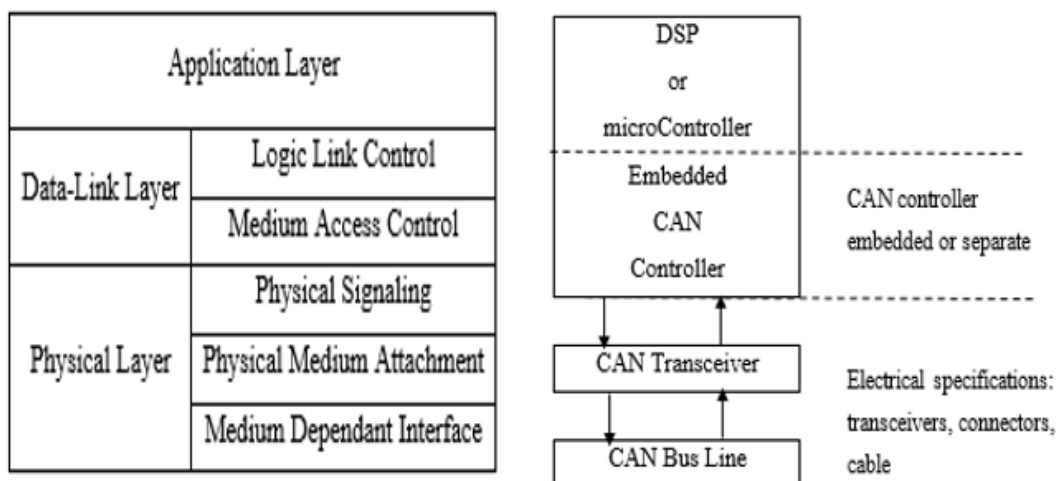


Figure 2. ISO11898 Architecture (Github 2018).

Choosing between the Standard or Extended CAN

Message Frames are used to transmit and receive data in the CAN system. The Message frames carry data from a transmitting node to one, or more, receiving nodes. The Message Frame formats supported by CAN protocol are the Standard CAN (CAN 2.0A) which uses 11-bit identifiers and the Extended CAN (CAN 2.0B) which uses 29-bit identifiers. The “standard” 11-bit identifier, providing 2^{11} or 2048 different message identifiers and the “extended” 29-bit identifier, providing 2^{29} or 537 million identifiers. However, both provide signaling rates from 125kbps to 1Mbps. (Texas Instrument 2016.)

Standard CAN (CAN 2.0A) 11-bit identifiers.

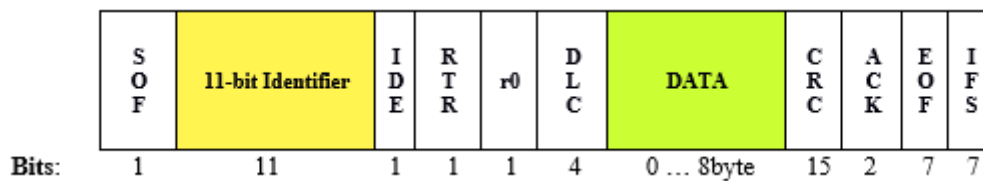


Figure 3. CAN 2.0A - Standard CAN Frame 11-Bit Identifier.

The standard CAN frame in figure 3 consists of the following bit fields: SOF – Start of Frame, Identifier – the standard CAN 11-bit identifier, RTR – Remote Transmission Request (RTR), IDE – Identification extension (IDE), r0 – Reserved bit, DLC – data length code, Data – allows up to 64bits (8bytes) of data to be sent, CRC – 16-bits (15-bits plus delimiter) cyclic redundancy check (CRC) containing the checksum used for error detection, ACK – Acknowledge bit, EOF – End of Frame bit has 7-bits and marks the end of a CAN frame (message) and disables bit stuffing and IFS – 7-bits interframe space bit contains the time required by the controller to move a correctly received frame to its appropriate position in a message buffer area. (Texas Instrument 2016.)

Extended CAN (CAN 2.0B) 29-bit identifiers.

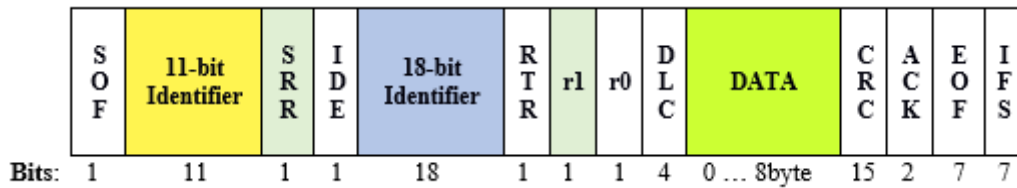


Figure 4. CAN 2.0B - Extended CAN Frame 29-Bit Identifier.

The Extended CAN in figure 4 is the same as the standard CAN message in figure 3, however, the Extended CAN message has additional bit fields such as: SRR – Substitute remote request (SRR) bit. It replaces the RTR bit in the standard message location as a placeholder in the extended format, IDE – When we have a recessive bit in the identifier extension (IDE), this implies that additional identifier bits follow the IDE of the 11-bit identifier, that is, the 18-bit extension which follows the IDE. It is an additional reserve bit included ahead of the DLC bit. (Texas Instrument 2016.)

Most CAN 2.0A controllers transmit and receive only Standard format messages, although some (known as CAN 2.0B passive) will receive Extended format messages but then ignore them. However, CAN 2.0B controllers can send and receive messages in both formats. (Texas Instrument 2016.)

The CAN Message Frame format used in this thesis was determined by the smart NOx sensor used. The smart NOx sensor has an Extended CAN ID of 0x18FEDF00.

2.1.3. CAN Messages

CAN messages can be said to be contents-addressed, that is, the content of the message implicitly determines their address. The messages are short – maximum utility load of 94 bits with no explicit address in the message. (Kvaser 2018a.)

CAN transmits message signals on the CAN network using two wires, CAN-High and CAN-Low. In a scenario of several sensors (nodes) need to send their data, the CAN bus implements a message priority identifier. The message with higher priority (lower binary message identifier number) wins the bus access. The bus access is a random event-driven process and if two nodes try to occupy the bus simultaneously, access is implemented using a nondestructive, bit-wise arbitration. Nondestructive implies that the node that wins the bus access continues with its message transmission without the message being destroyed or corrupted by the other nodes. The priority allocation feature makes CAN to be attractive in its application to real-time control environment. (Texas Instrument 2016.)

CAN controller uses an arbitration process to handle the message transmission priority as each node continuously monitors its own transmissions. For example, in figure 5 node B's recessive bit is overwritten by node C's higher priority dominant bit and node B detects that the bus state does not match the bit that it transmitted, therefore it pauses its transmission allowing node C to continue with transmitting its message. Node B then makes another attempt to transmit its message when node C has completed its message transmission and the bus is free. This functionality is present entirely within the CAN controller as it is part of the ISO 11898 physical signaling layer and it is completely transparent to a CAN user. (Texas Instrument 2016.)

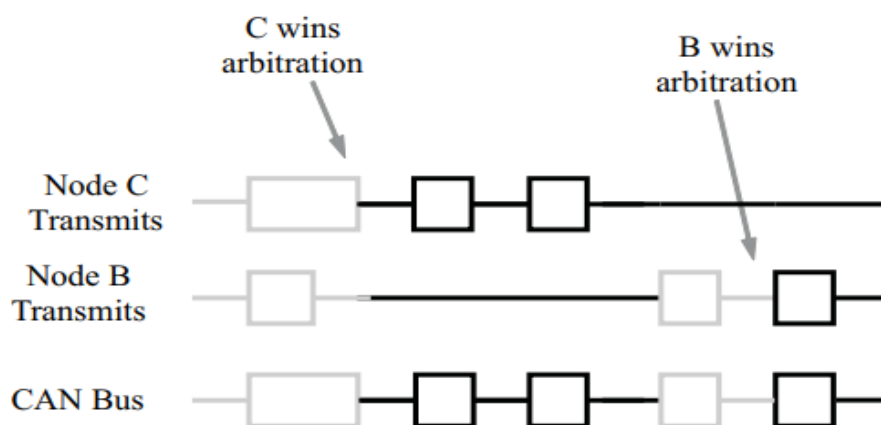


Figure 5. Arbitration on a CAN Bus (Texas Instrument 2016).

On a CAN bus, the CAN message/frames are of four types namely data frame, remote frame, error frame, and overload frame. They are not discussed here in details as they are not part of the scope of the research. (Kvaser 2018b.)

The CAN Bus is a reliable and robust bus because of its error handling capability. The CAN protocol uses five techniques of error checking. It uses three at the message level and two at the bit level. A message that fails any one of these error detection techniques is not accepted leading to the generation of an error frame from the receiving node. When this happens, the transmitting node is forced to retransmit the message until it is received correctly. However, for a faulty node that hangs up a bus when its continuously in error, its ability to transmit is disabled by its controller when an error limit is reached.

2.2. Wireless Communication Protocols

In this chapter, some available wireless solutions being used to connect remote sensors and devices to a central monitoring system are analyzed. These wireless solutions can be applied in several areas, however, selecting the right solution and using it in the right application is very crucial and can be a tough task having several associated risks.

All wireless communication comprises of the following components; a transmitter, receiver, antennas, channel, and the environment. The transmitter sends signals to an antenna for transmission and the radio transmitter encodes data in RF waves having significant signal strength (power output) to transmit the signal to a receiver. The receiver collects and decodes the data arriving at the receiving antenna. At the receiver, assigned RF signals are received and decoded while discarding the unwanted signals. Different radiation patterns are generated by antennas depending on their design and application. The antenna also has a gain which is a measure of how much energy is focused in a direction. (DIGI 2016.)

In describing the wireless communication environment or path, there are two types of LOS generally used namely Visual LOS – which is the ability to see from one point to the other. A straight linear path between two points is required. RF LOS – this need not

only visual LOS, but also requires a Fresnel zone (football-shaped path) that has no obstacles so that data can travel optimally from point A to point B. The Fresnel zone can be assumed to be a tunnel between two sites that provide a path for RF signals as illustrated in figure 6. (DIGI 2016.)

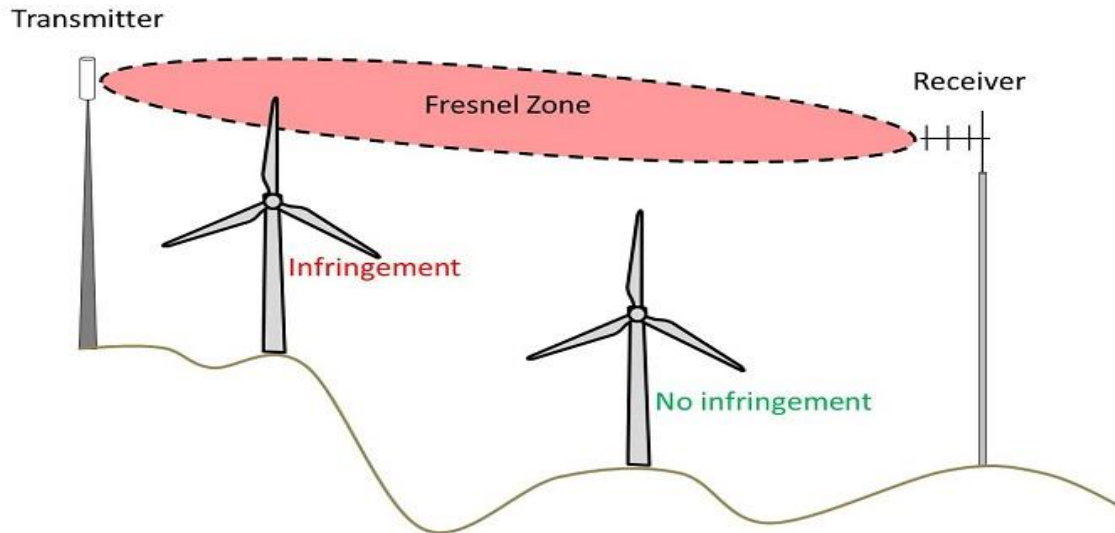


Figure 6. Fresnel zone illustration (Frolic 2016).

2.2.1. Factors that affect wireless communication

Wireless applications typically require burst transmission, reduced overhead, and they use a very small amount of data per node, therefore, the bandwidth is not the main requirement. Some applications require coverage of large areas; reliability, availability, bounded latency for real-time behavior and energy efficiency as some key performance indicators. (Khan & Turowski 2016.)

Industrial environments differ significantly when compared to the office and home environments. Certain challenges exist like high temperatures, very high airborne particulates, multiple obstacles and long distances between equipment and systems, making it hard to place and get access to sensors, transmitters, and other data communication devices. These and several other factors make setting up of data communication channels that is reliable, long-lasting, and cost-effective, a rare, complex, and costly challenge.

From past surveys, according to B&B Electronics, for several reasons such as noise, channel interference, and signal echo etc, wireless I/O has typically not performed well enough to endure the harsh demands of industrial applications (Advantech B+B SmartWorx 2018.)

The typical open radio frequencies such as the 900 MHz and 2.4 GHz are used in recent wireless data communication applications and can go through office cubicles walls, drywall, wood and other materials which are found in homes or offices. However, they are usually deflected by larger objects, metals, and concrete. As a result, it can change the data signal path returning it to the original transmitter and thereby resulting in an “echo” or “multi-path”. In the first-generation wireless systems, this led to the cancellation of the transmission as the system becomes confused with this type of bounce interference. This resulted in a state called “radio null” and prevents data communication. In the case of noise, large motors create electromagnetic emissions while heavy equipment, high power generation, and usage, and other typical industrial machinery can generate very high levels of “noise” which in turn interferes with early wireless equipment. In these “noisy” environments, transmitters and remote nodes were unable to communicate with each other, resulting in frequent data loss. (Advantech B+B SmartWorx 2018.)

The radio frequency space becoming very crowded has led to the challenge of channel sharing and interference. This means that the frequency spectrum approved by the FCC were shared amongst many devices, which includes the devices using IEEE 802.11 and IEEE 802.15.4. This resulted in frequent data mix up as receivers and nodes received and transmitted information on the same channel as the other devices in the area. The wide distances between the central control systems and remote sensors made it not feasible for the early wireless systems with ranges of several hundred feet or more to allow communication. The era of wireless communication also created many security issues and it continues to require a high level of counter-measures to ensure the safety of data and business systems. (Advantech B+B SmartWorx 2018.)

There are modulation and transmission schemes that have been developed to cater for the effects of these challenges and interference. The two most optimum to look for are FHSS (Frequency Hopping Spread Spectrum) which requires narrow bandwidth. In this scheme, data is transmitted through a single channel at a time, but the channel is constantly and rapidly changing or hopping. However, for DSSS (Direct Sequence Spread Spectrum), this scheme requires large bandwidth. Data is transmitted simultaneously over every available channel, this makes it a bit more reliable in noisy environments. (Advantech B+B SmartWorx 2018.)

2.2.2. Types of Wireless Communication Protocols

It is important to take caution when designing wireless networking systems, all wireless transmitters, nodes and equipment must support the same transmission scheme. There are many proven wireless standards out there that can be implemented and developed into a design that takes into consideration the features like signal reliability, security, distance, speed, and efficiency. Trying to find out the best solution would depend on where it is to be applied and the needs involved. The wireless protocols available has its uses and advantages. Identifying the one that suits your application in a given industrial application begins with finding the best match for packet delivery rate, number of devices, distance, data rates, cost, power consumption, and most importantly reliability and security. (DIGI 2016.)

There are different communication technologies aimed at low power and wireless IoT communication and there are categorized into two namely:

Low Power Local Area Networks which has less than 1000 meters range. This category includes IEEE 802.15.4 (for example, ZigBee), WiFi and Bluetooth/BLE, etc., applicable directly in short-range personal area networks, body area networks and if well organized in a mesh topology, also in larger areas.

Low Power Wide Area Networks has a greater coverage range than 1000 meters are essentially low-power versions of cellular networks, with each “cell” covering thousands

of end-devices. These include LoRa (LoRaWAN), and protocols, like Sigfox, DASH7, etc.

Sections 2.2.3 to 2.2.6 presents some of the most industrially relevant wireless protocol options with some corresponding pros and cons.

2.2.3. Bluetooth Low Energy (BLE)

Bluetooth is first briefly discussed before presenting the BLE protocol. Bluetooth wireless communication protocol technology with is a short-range and a frequency range of 2.4 to 2.485 GHz made as a substitute for wired connections and applied in many devices such as headphones, and speakers, etc.

It was created by Ericson Mobile in 1994 as a substitute for wired cables and its spread spectrum technology is frequency-hopping based. This also means that devices keep their link preserved even when there is no data flow. When the device goes to sleep it is in Sniffer mode which reduces power consumption and provides up to several months of battery life even at Peak transmit current of typically around 25mA. Bluetooth consumes a significantly small amount of power than other radio standards, but it is however not low enough for smaller battery cells like the coin battery cells and energy harvest- ing applications. (Bluetooth SIG 2018a.)

Bluetooth Low Energy is a short-range wireless protocol used for applications that does not require handling large amounts of data (throughput) and can therefore remain on battery power for years. BLE is made to provide considerably reduced power consumption, and low cost while maintaining very similar communication range to standard Bluetooth; otherwise known as, radio coverage. However, BLE is not backward-compatible with previous Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR) protocol sometimes referred to as "classic". The Bluetooth 4.0 specification permits devices to implement either or both LE and BR/EDR systems. (Adafruit Industries 2018.)

BLE does not have data throughput because BLE does not support streaming data. When a connection has been established (paired), BLE spends most of the time in sleep

mode waiting to send or receive the next set of device status information referred to as ‘expose state’, such as the Battery Level. It has a data rate of 1Mbps which allows for quick data transfer of small chunks or data packets (kB), exposing the state of the device to retrieve the information. The status update interval rate delay can be programmed from 7ms up to 4s between data polls. Once data has been transferred, a few milliseconds, the BLE goes back to sleep to conserve battery; whereas, standard Bluetooth stays on the entire time even when information is not being transferred. (Adafruit Industries 2018.)

The main features of BLE that differ from standard Bluetooth are described in table 1.

Table 1. Main Features of BLE that differ from standard Bluetooth.

Features	Details
The PHY or physical layer	has parts that were derived from the Bluetooth Radio
Advertising	altered to simplify the discovery and connection
Asynchronous connection-less MAC	used for fast transactions with low latency, (e.g. 3ms from start to finish)
Generic Attribute Profile (GATT)	has been simplified between the devices and software
Asynchronous Client / Server architecture	redesigned to have the lowest cost and ease of implementation
BLE was designed for exposing state of devices and retrieving the information	data can be read at any time by a client, such as a Smartphone App; it’s good at small, discrete data transfers and data can be triggered by local events

Figure 7 shows a graphical representation of the frequency spectrum used on BLE.

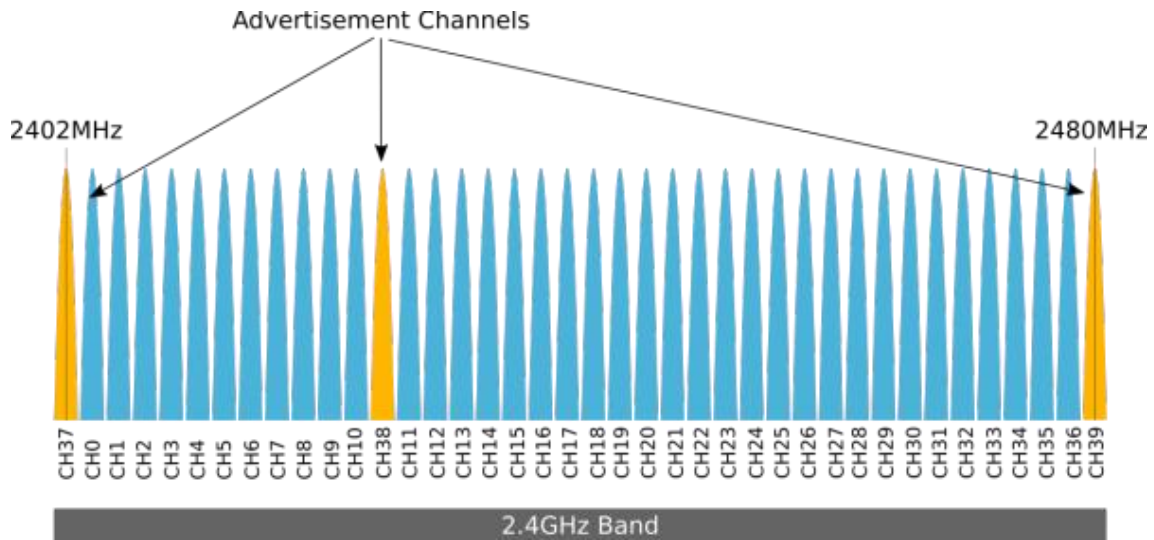


Figure 7. Bluetooth Low Energy Frequency Channels (Argenox Technologies 2018).

BLE Security - Bluetooth Core Specification provides several features to ensure data encryption, data integrity and data privacy. The first feature is a pairing mechanism in which the devices participating in the communication exchange information about their identity to set up trust and prepares an encryption keys for future data exchange. The second feature is the public/private key generation which is performed by the Host on each low energy device independent of any other device and each device involved in pairing contributes to the generation of the Secure Connection Key. BLE uses the third feature called AES-CCM cryptography which generates a 128-bit data encryption algorithm for the encryption of data. The fourth feature is the signed data where BLE uses a Message Authentication Code generated by the signing algorithm and a counter to securely send authenticated data over an unencrypted communication channel. Lastly, the fifth feature is privacy in which the ability to track a LE device over a period of time is reduces as a result of the frequent changing address of the BLE device. This frequently changing address is referred to as the private address and it can be resolved by the trusted devices. (Bluetooth SIG 2018b.)

Some essential BLE Radio features are described in table 2.

Table 2. BLE Radio feature.

Features	Description
Range	~150m open field. Increased modulation index provides a larger range > 100m
Output Power	~10mW (10dBm)
Max Current	~15mA
Latency	allows an application to form a connection and then transfers the authenticated data within a few milliseconds
Topology	Star configuration allows for one-to-many connections
Data Transfers	data packets (8 octet min up to 27 octets max) are transferred at 1 Mbps
Connections	> 2 billion devices use a 32-bit access address on every packet
Modulation	GFSK @ 2.4 GHz ISM Band for all Data Transfers
Robustness	Adaptive Frequency Hopping, 24-bit CRC on all packets ensuring the robustness
Security	128bit AES CCM provide strong encryption and authentication of data packets
Sleep current	~ 1 μ A
Modes	Broadcast, Connection, Event Data Models Reads, Writes
Sniffer	advanced sniff-sub rating achieves ultra-low duty cycles, conserving battery life

Pros – This wireless solution has a lower power requirement in the market compared to other design such as the WiFi, LoRa and ZigBee. It also has, when compared, the lowest cost, and perhaps has the fastest development platform available. **Cons** – Since it is designed for low energy, the communication rate was not a factor in the design, so information is only transmitted in small bursts of data; of course, this could be considered a ‘Pro’ or an advantage depending on the specific use of this technology. (Advantech B+B SmartWorx 2018.)

2.2.4. Zigbee (IEEE 802.15.4)

Another short-range wireless protocol is the ZigBee, which is a standard for personal-area networks developed by ZigBee Alliance aiming at providing a low cost, low power consumption, reliable and two-way wireless communication standard for short-range applications. ZigBee is a decentralized network which is very similar to the internet and having support for self-healing mesh networking. It allows the nodes to find new routes throughout the network when one route fails, thereby making it a robust wireless solution. (Texas Instrument 2013.)

ZigBee was designed by ZigBee Alliance with the purpose of providing low-cost, low-power consumption, two-way and reliable wireless communication standard for short-range applications. It is a personal area network standard that is completely open and was ratified by the Institute of Electrical and Electronics Engineer (IEEE) in 2003. It has a protocol stack based on the IEEE 802.15.4 standard and has advantages such as long battery lifetime, supports many nodes (up to 65000) in a network, ease of deployment, low-cost, and global usage. (ZigBee Alliance 2012.)

The ZigBee stack architecture has four layers namely Physical Layer (PHY), Medium Access Control Layer (MAC), Network Layer (NWK) and Application Layer (APL).

Each layer is applied to a specific set of services for the previous layer above. A data entity provides a data transmission service and a management entity provides all other services. The first two layers namely the Physical Layer (PHY) and Medium Access Control Layer (MAC) are defined by the IEEE802.15.4-2003 standard, while the Network Layer (NWK) and the frame for the application layer, which consist of the Application Support sub-layer (APS) and the ZigBee device objects (ZDO), are built by the ZigBee Alliances. (ZigBee Alliance 2012.)

Figure 8 shows the ZigBee packet structure.

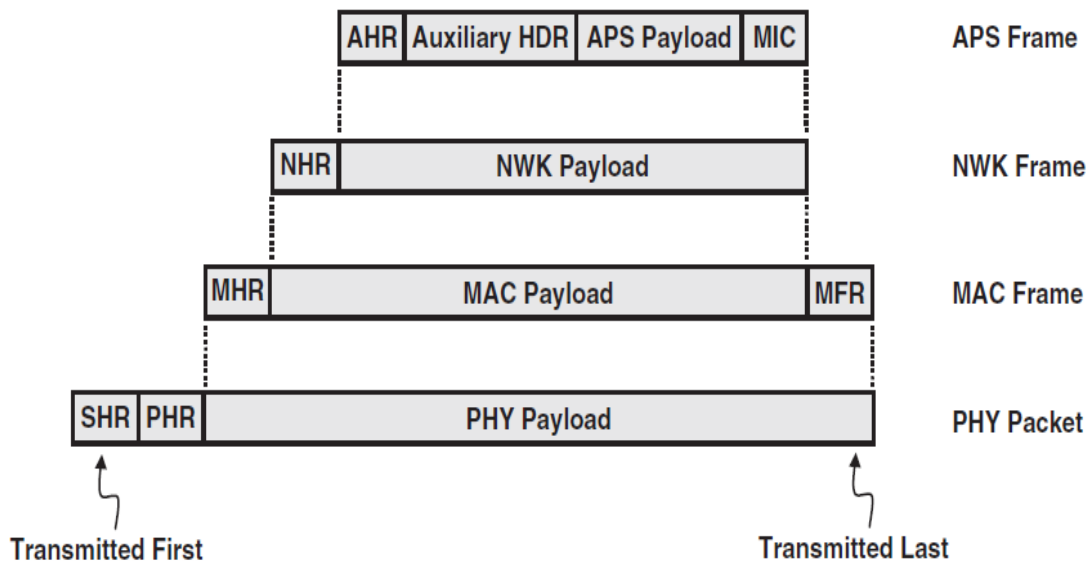


Figure 8. Zigbee Packet Structure (Zybuluo 2018).

ZigBee operates on two separate frequencies ranges, 868/915 MHz and 2.4 GHz. The lower frequency PHY layer covers the 868 MHz European band and the 915 MHz band which is used in counties like the United States and Australia. The higher PHY layer frequency is used worldwide. (ZigBee Alliance 2012.)

ZigBee protocol supports 3 nodes types namely ZigBee Coordinator ZC, ZigBee Router (ZR) and ZigBee End Device (ZED). The ZC initiates the network, protects it and generates the control functions needed. After the initiation of the network, the PAN coordinator works as a ZigBee Router (ZR). If the network is operating in the beacon-active mode, the ZC periodically sends beacon frames to be able to synchronize the rest of the network. While in cluster free topology, all the ZRs receive beacons from their parents and sends their own beacons to the nodes in their cluster. The ZR directs the data detected to the sink node. It can perform a multiple node hooping role and does this by having a relation to the ZC or ant previous ZR. The ZED serves one purpose only and that is, being normal nodes without any routing features. (Vançin & Erdem 2015.)

Figure 9 shows the outline of the ZigBee stack architecture.

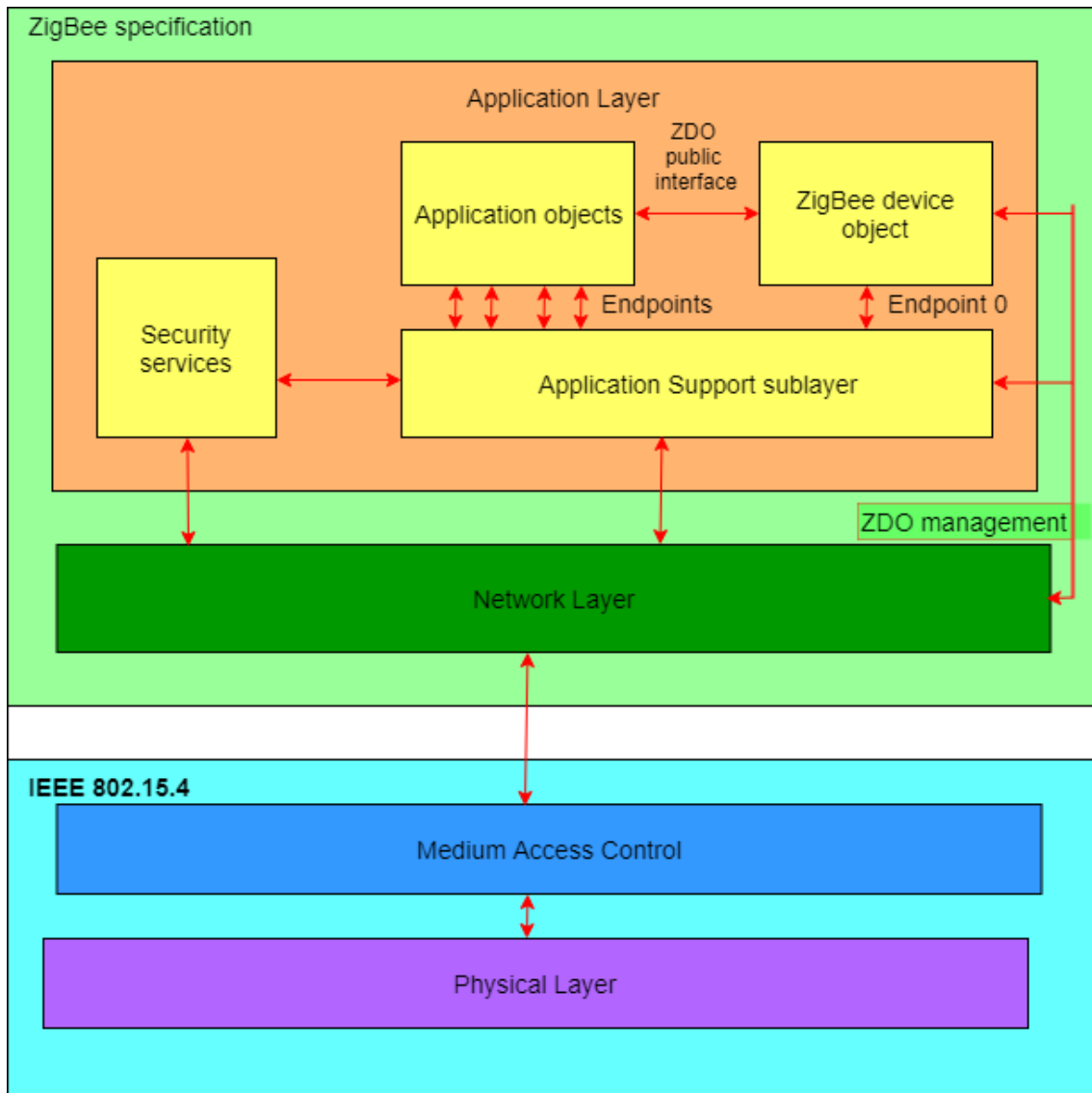


Figure 9. ZigBee Protocol Stack Architecture (3dfury 2012).

The topologies used by ZigBee are star, tree and mesh as shown in figure 12. The tree topology in figure 12 is suitable for wireless sensor networks due to its low power consumption and cost. Its power protection process is provided by the IEEE802.15.4/ZigBee Mac frame. However, it has drawbacks related to restrict routing process and band usage and any disconnection in the tree topology bring delay in data flow and a heavy workload is created in the recovery process. This topology is better than mesh topology with respect to usage of memory since a single rout is used from the

source node to the destination node and the excess memory is not saved. (Vançin & Erdem 2015.)

The star topology has a communication structure that is centrally managed with its architecture based on a central node. The ZEDs do not interact with each other directly but communicate with each other through the ZC in the center. The ZC has a PAN ID that is not defined in any other ZigBee network in the environment. However, since the star topology consumes battery power rapidly because it points towards the center and the ZigBee clustering is cumbersome while addressing large-scale networks, it is not suitable for wireless sensor networks. The mesh topology is more power efficient when using batteries than the star topology. It is a centralized structured topology where any node can reach other nodes in the network and communicate directly, thereby, giving the network high flexibility but also introduces the complexity of end-to-end communication. (Vançin & Erdem 2015.)

ZigBee finds its application in the following areas such as Building Automation, Health Care, Home Automation, Input Devices, Remote Control, Retail Services and Smart Energy and Telecom Services. (ZigBee Alliance 2012.)

ZigBee Security

The three security modes supported by ZigBee standard are residential security which requires a network key to be shared among the source and destination devices, the standard security which adds several optional security enhancements over the residential security, including an APS layer link key and the high security which adds entity authentication and other features not widely supported.

ZigBee security is divided into two levels. The application layer security and the network layer security. The AES-128-bit encryption algorithm is used for the security. The security is used to ensure message integrity, confidentiality and entity authentication. (Mukherji & Sadu 2016.)

Application layer security - The APS layer security is used to encrypt the application data using a key that is shared between source and destination devices. APS security is optional and provides end-to-end security using APS key that is known only to the source and destination devices, whereas, network layer security is applied to all the data transmission and is decrypted and re-encrypted on a hop-by-hop basis. When the APS security is enabled, the data are encrypted as shown in figure 10 below. (DIGI 2018.)

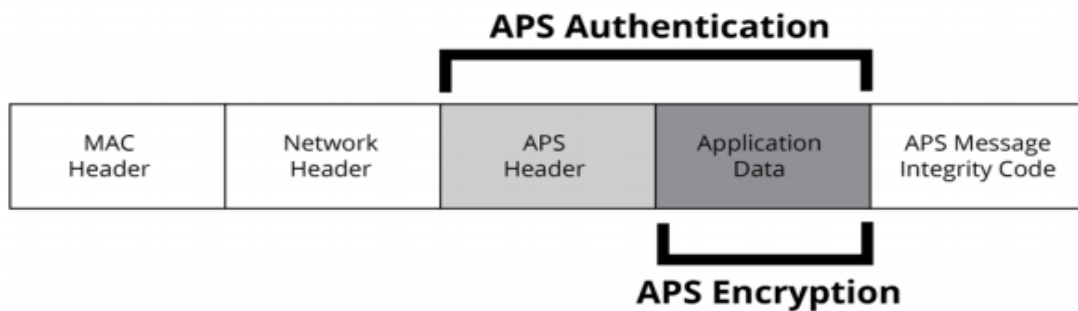


Figure 10. Application Layer Security (DIGI 2018).

Network layer security - The network key is used in encrypting the APS layer and application data. Apart from encrypting application messages, network security can also be applied to route request and reply messages, APS commands, and ZDO commands. However, network encryption is not applied to MAC layer transmissions such as beacon transmissions. When you enable security on a network, all the data packets are encrypted with the network key as shown in figure 11 below. (DIGI 2018.)

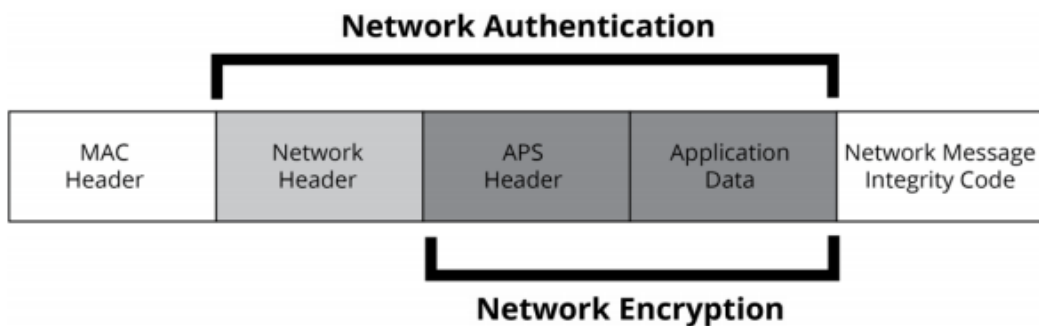


Figure 11. Network Layer Security (DIGI 2018).

The packets encrypted by network layer key are encrypted and decrypted by each hop in the network. On receiving a packet with network encryption, the receiving device will decrypt the packet and authenticate the packet. If the device is not the expected destination, it encrypts the packet using its details and sends to the next hop. (DIGI 2018.)

Application and Network layer security - Applying both application and network layer security at the same time is possible. Figure 12 demonstrates the authentication and encryption performed on the final Zigbee packet when both are applied. (DIGI 2018.)

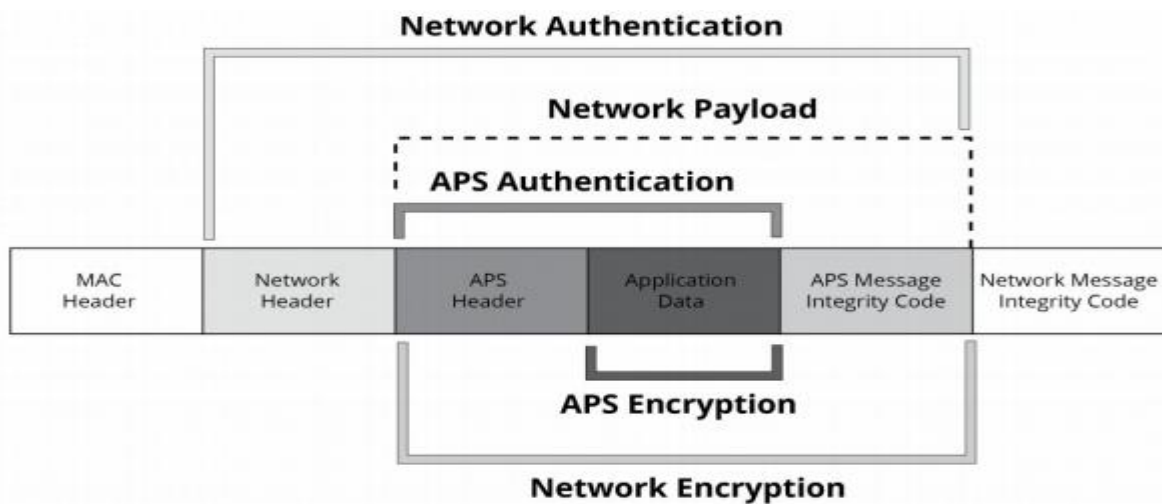


Figure 12. Application and Network Layer Security (DIGI 2018).

Pros – It is much more power efficient when compared to WiFi and Bluetooth as a result of its advanced sleep and sniffs capabilities. It operates with an even smaller physical footprint than Bluetooth and has a higher penetrating power. **Cons** – ZigBee's poor interoperability is a disadvantage as well as its low data rate of 720 kbit/s. It is relatively unpopular and efforts are still being made by hardware developers to improve its architecture. (Advantech B+B SmartWorx 2018.)

2.2.5. WiFi (IEEE 802.11 b/g/a)

Wireless fidelity (WIFI) is a wireless networking technology which utilizes radio waves to provide a wireless high-speed internet and network connections.

The IEEE 802.11 (b/g/a) standards are presented as follows. The IEEE 802.11b has an operating frequency of 2.4GHz radio spectrum with a range of 100 -150 feet. It is the most popular and least expensive. Since 802.11b uses the same unregulated radio signaling frequency (2.4 GHz) as original 802.11 standard 802.11b devices can have interference from other appliances using the same 2.4 GHz range such as microwave ovens and cordless phones, etc. However, when you install 802.11b devices with an adequate distance from other appliances, the interference can easily be avoided. The IEEE 802.11a standard is less popular and has an operating frequency of 5GHz with a shorter range of 50 -75 feet due to its higher frequency. It is more expensive and is not compatible with 802.11b. 802.11a supports bandwidth up to 54 Mbps. Its higher frequency also implies that 802.11a signals penetrate walls and other obstructions with more difficulty. IEEE 802.11g combines the features of both 802.11b and 802.11a with a range of 100 - 150 feet and operates at a radio frequency of 2.4GHz. It is compatible with 802.11b. (Symmetry Electronics 2018.)

When connected to the internet, WiFi gives a full TCP/IP stack. The integration of WiFi to most technologies of today such as laptops, smart phones, tablets and TVs makes it a well-established standard. Most WiFi networks operate on the 2.4 GHz band. It has a capability of operating at 5 GHz giving clearer signal with more channel space. However, the range of 5 GHz is shorter than 2.4 GHz, which is why the 2.4 GHz is often used in homes. Power consumption of WiFi has been an issue making it not efficient for IoT devices, however, this issue can be negligible when the WiFi module is combined with a powerful microprocessor making it capable of consuming power less than other modules like the 433 MHz. (Darshana, Wilkie & Irvine 2016.)

WiFi is not usually utilized in the building IoT nodes due to its high-power consumption interfaces. It consumes 40 times the power during transmission and 10 times more than a Bluetooth Low Energy node when receiving. New technologies like WiFi and 4G-LTE internet access has contributed to the growth of the information communication network. The IP addresses and the domain names are the fundamental assets of Internet used to identify and get the position of the networking equipment in the Inter-

net. However, services like information retrieval become important infrastructure of Internet to maintain all applications of Internet. (Walia, Kalra, & Mehrotra 2016.)

Three main reasons why WiFi networks do not support sensor networks sufficiently are first lack of power saving mechanisms - The peculiar energy constraints of sensor networks are not considered in the IEEE 802.11 standard; energy saving mechanisms specially designed for these types of devices are not included in the standard, secondly using unsuitable bands - Based on their short wireless range and high obstruction losses, current WiFi bands need to make use of intermediate nodes which makes the network more complex. Implicitly, this means that there is a lack of an implementation of a band in the IEEE 802.11 standardized that will be suitable for low-rate and long-range networks and lastly, availability of low-cost alternatives - Due to the low usage of WiFi for data communication between low-capability and battery-powered nodes, there has been a rise in the development of low power alternatives such as IEEE 802.15.4, 6LoWPAN, Zigbee, and sub-1GHz proprietary protocols, all referred to as WSNs. (Adame, Bel, Bellalta, Barcelo & Oliver 2014.)

WiFi finds its application in several areas such as military and aerospace, medical electronics, network and server equipment, automotive car electronics, industrial and home networking and mobile phones, etc.

WiFi Security - The WiFi network security requirements can be categorized into three main components, first is authentication which involves user authentication and server authentication and second is integrity involving the maintenance of the accuracy and consistency of data and the third is privacy. Security ensures message integrity and confidentiality. WiFi network makes use of certain encryption algorithms to provide security, allowing the control of who connects, and privacy, preventing unauthorized persons to read the transmitted data. During wireless communication, to ensure maximum security the network should include only devices with the latest security technology. It can use the AES-128-bit encryption algorithm to provide security. Others include SSL3/TLS1, HTTPS, RSA, AES-256, 3DES, RC-4, SHA-1, MD-5, WEP, WPA and WPA2 accelerated in hardware: AES, 3DEC and SHA. (Lin 2014.)

Pros – This is the typical method of networking for businesses, homes, and offices. WiFi is widely used for its high data transfer rates between 12MB/s up to 54 MB/s. It provides advantages like mobility, ease of installation, flexibility, cost, reliability, security, use unlicensed part of the radio spectrum, roaming and speed. **Cons** – However, complying with this standard requires excessive overhead in relations to power consumption, processor resources, short range (160m max), software, and the physical component size, making it less than effective in most situations. (Advantech B+B SmartWorx 2018.)

2.2.6. LoRa (Long Range)

LoRa is a “Long Range” wireless communication protocol marketed by LoRa Alliance. LoRaWAN uses the MAC layer protocol to provide a medium access control mechanism which enables many end-devices to communicate with a gateway making use of a proprietary LoRa modulation. However, the LoRaWAN is an open standard that is being developed by LoRa Alliance. LoRa is a new, private spread-spectrum modulation technique that allows sending data at extremely low data rates to extremely long ranges. The low data rate, which goes down to few bytes per second, and LoRa modulation lead to very low receiver sensitivity as low as -134dBm, which when combined to an output power of +14dBm implies extremely large link budgets of up to 148dB. This implies more than 22km (13.6 miles) in LOS links and up to 2km (1.2miles) in NLOS links for urban environment which can go through buildings. LoRa uses the Sub-1 GHz spectrum, that is, the 900MHz ISM band in the U.S. and the 868MHz ISM band in Europe, to provide the long-range connectivity. (LoRa Networking Guide 2017.)

LoRa was originally designed for IoT slow sampling rate, long distance communication. The LoRaWAN defines the Data Link (DL) layer above the Physical Layer (PHY) defined by LoRa radio. LoRaWAN has a good scalability, cellular architecture and central coordination function. These two-parted systems can work together when several sensor nodes are involved. The physical layer is implemented using LoRa that exploits the Chirp Spread Spectrum (CSS) modulation using specialized transceivers. The chirp symbol can encode a variable number of bits represented by Spreading Factor (SF). A

Forward Error Correction (FEC) is also implemented as a Hamming Code H (M, K) where $M = \{5, \dots, 8\}$ is the codeword length and $K=4$ is the block length. The LoRaWAN defines the coding rate as $CR=K/M$ and the typical chirp bandwidth in the 868 MHz band is $B [125, 250]$ kHz; but the spreading factor varies from SF [7, 12]. (Rizzi, Ferrari, Flammini, Sisinni & Gidlun 2017.)

LoRa was defined to provide a variable chirp duration T_c as seen in equation 3 and BW is not affected by the SF, therefore, the raw bit rate R_b can be computed using equations 1 and 2.

$$T_c = \frac{2^{SF}}{BW} \quad (1)$$

$$R_b = SF * \frac{BW}{2^{SF}} * \frac{K}{M} \quad (2)$$

where R_b is the raw bit rate, SF is the spreading factor, BW the bandwidth, K the block length, and M the codeword length.

The Value of R_b can vary from 366 bps (BW=125 kHz and SF=12) to 11 bps (BW=250 kHz and SF=7). One thing to note is that different SF are pseudo-orthogonal, meaning that packets using SF=i and SF=j can still be decoded even if they overlap in time and frequency provided that $i \neq j$ and the received packet's signal to Interference plus Noise Ratio (SINR) is above the isolation threshold which is a function of I and j. These parameters affect the decoder sensitivity. An increase in bandwidth lowers the receiver sensitivity, whereas, an increase of the spreading factor increases the receiver sensitivity.

When the code rate is reduced, the Packet Error Rate (PER) also reduces when there is a short outpour of interference, that is, a packet transmitted with a code rate of 4/8 will tolerate interferences more than a signal transmitted with a code rate of 4/5. Table 3 taken from the SX1272 datasheet shows the device Variants and Key Parameters. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

Table 3. LoRa Device Variants and Key Parameters taken from LoRa SX1272/73 Datasheet, Rev. 3.1. Semtech, 2017.

Part Number	Frequency Range	LoRa™ Parameters			
		Spreading Factor	Bandwidth	Effective Bitrate	Sensitivity
SX1272	860 – 1020 MHz	6 - 12	125 – 500 kHz	0.24 – 37.5 kbps	-117 to -137 dBm

The LoRa symbol rate R_s is defined in equation 3 as:

$$R_s = \frac{1}{T_c} = \frac{BW}{2^{SF}} \quad (3)$$

Where T_c is the chirp duration, BW is the programmed bandwidth and SF the spreading factor. The transmitted signal is a constant envelope signal. Equivalently, one chip is sent per second per Hz of bandwidth.

LoRa Packet structure and Payload

The LoRa™ modem uses two types of packet format namely the explicit and implicit formats. The explicit packet includes a short header that contains information about the number of bytes, coding rate and whether a CRC is used in the packet. Figure 13 shows the LoRa packet structure. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

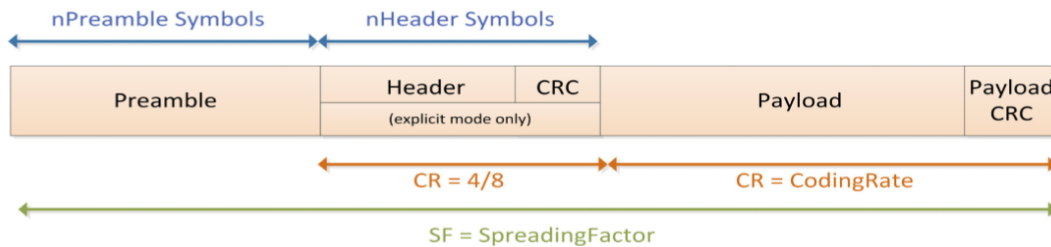


Figure 13: LoRa Packet Structure (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1).

The three elements of the LoRa packets are:

A preamble - The preamble is used in synchronizing the receiver with the incoming data flow. The default configuration of the packet is a 12-symbol long sequence. This is programmable to make the preamble length extendable in applications where reducing the receiver duty cycle is needed in receive intensive applications. The transmitted preamble length is adjusted using the registers *RegPreambleMsb* and *RegPreambleLsb* from 6 to 65535 with total preamble lengths of $6 + 4$ to $65535 + 4$ symbols once the overhead of the preamble data is considered. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

An optional header - The header type is dependent on the mode of operation chosen, the header type is selected using the *ImplicitHeaderModeOn* bit found within the *RegModemConfig1* register. The Explicit header mode is the default header mode and we also have the Implicit header mode. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

The data payload - The packet payload of LoRa is a variable-length field that contains the actual data coded at the packet error rate either as specified in the header in explicit mode or in the register settings in implicit mode. An optional CRC may be appended to it. Using a given combination of spreading factor (SF), coding rate (CR) and signal bandwidth (BW), the total on-the-air transmission time of a LoRa packet can be calculated as illustrated below. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

The definition of the symbol rate leads to the definition of the symbol period in equation 4.

$$T_s = \frac{1}{R_s} \quad (4)$$

However, the LoRa packet duration is the sum of the duration of the preamble and the transmitted packet. Where the preamble length is computed as in equation 5.

$$T_{preamble} = (n_{preamble} + 4.25) * T_{sym} \quad (5)$$

where n_{preamble} is the programmable preamble length, taken from the register *RegPreambleMsb* and *RegPreambleLsb*. The payload duration is dependent on the header mode that has been enabled. The number of payload symbols is given by the equation 6.

$$n_{\text{payload}} = 8 + \max\left(\text{ceil}\left[\frac{8\text{PL} - 4\text{SF} + 28 + 16\text{CRC} - 20\text{IH}}{4(\text{SF} - 2\text{DE})}\right](\text{CR} + 4), 0\right) \quad (6)$$

where PL is the number of bytes of payload, SF is the spreading factor, IH = 1 when implicit header mode is enabled and IH = 0 when explicit header mode is enabled. When DE is set to 1, it indicates the use of the low data rate optimization, while 0 indicates its disabled. CRC shows the presence of the payload; CRC = 1 when on and 0 when off. CR is the programmed coding rate from 1 to 4. The ceil function indicates that the portion of the equation in square brackets should be rounded up to the next integer value. While the max function compares the evaluated ceil value result and returns 0 or the result depending on which one is higher. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

$$T_{\text{payload}} = n_{\text{payload}} * T_s \quad (7)$$

Equation 7 is used to compute the total payload. Therefore, the total on-the-air transmission time of a LoRa packet is the addition of the preamble duration and payload duration as shown in equation 8.

$$T_{\text{packet}} = T_{\text{preamble}} + T_{\text{payload}} \quad (8)$$

According to the LoRa SX1272/73 Datasheet, Rev. 3.1. Semtech, 2017, the LoRa module utilizes frequency hopping spread spectrum (FHSS) typically used when the duration of a single packet could exceed the regulatory requirements relating to the maximum allowed channel retention time. This is, however, most noticed in the case of the US operation where the 902 to 928 MHz ISM band which makes provision for frequency hopping is used. LoRa modem enables the FHSS by setting the FreqHoppingPeriod bit to a non-zero value in the register RegHopPeriod. The time in which the transmission will dwell in any channel is determined by the FreqHoppingPeriod which is an “integer” multiple of the symbol periods as illustrated in equation 9.

$$\text{HoppingPeriod}_{[s]} = T_s * \text{FreqHoppingPeriod} \quad (9)$$

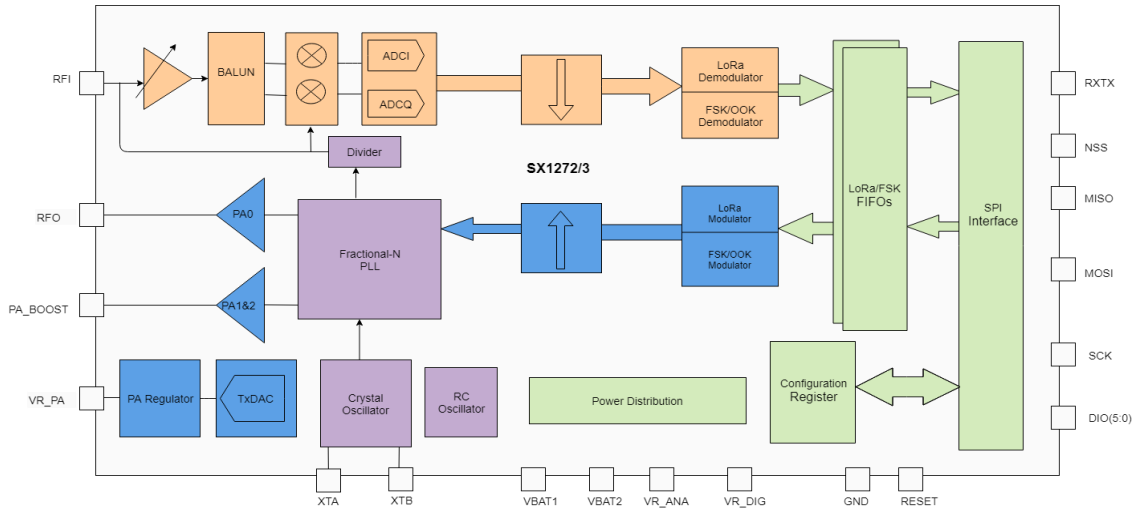


Figure 14. A Simplified SX1272 Block Diagram (SX1272 LoRa Datasheet 2017).

A Simplified SX1272 Block Diagram is illustrated in figure 14. LoRa and LoRaWAN can meet the requirements of industrial environments particularly when application scenario needs cycle time in the order of one minute and for a large number of sensor nodes. LoRa is focused on applications where the end devices have limited energy (battery-powered) and where end devices do not require transmission of more than a few bytes of data at specific time and where the initiation of data traffic can be done by either the end-device (for example, when the end-device is a sensor) or by an external entity that wants to communicate with the end-device (like when the end-device is an actuator). (LoRa Alliance White Paper 2015.)

LoRa physical layer which was developed by Semtech operates on the 433, 868 and 915 MHz ISM band depending on the region in which it is to be deployed. In Europe, the 868 MHz ISM band is used. The payload on each transmission can vary from 2 to 255 octets and the data rate reaches up to 50Kbps when the channel aggregation is employed. The modulation technique used is proprietary to Semtech. LoRaWAN gives a medium access control mechanism, enabling many end-devices to communicate with the gateway using the LoRa modulation. The LoRaWAN is an open standard being de-

veloped by LoRA Alliance unlike the LoRa modulation which is proprietary. (Semtech SX1272 LoRa Datasheet 2017, Rev. 3.1.)

The typical LoRa network uses “star-of-stars” topology as seen in figure 15.

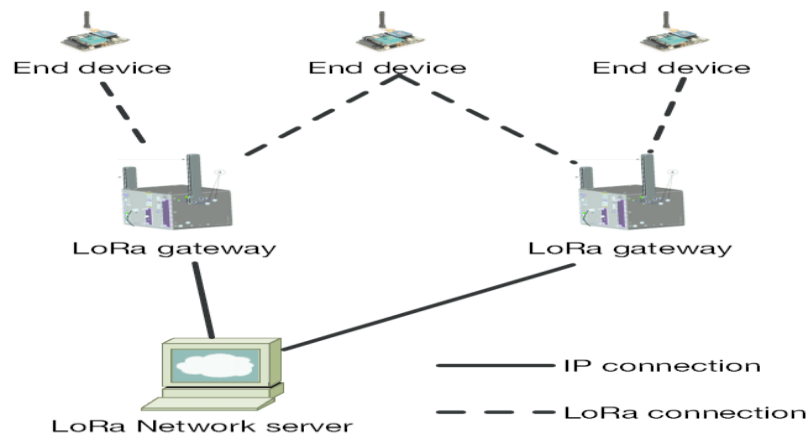


Figure 15. LoRa Network Architecture (ResearchGate 2018).

From figure 15, the end-devices communicate with gateways using LoRa with the LoRaWAN. The gateways forward raw LoRaWAN frames from the devices to a network server over a back-haul interface with a higher throughput, using Ethernet or 3G. Consequently, gateways are only bidirectional relays, or protocol converters, with the network server being responsible for decoding the packets sent by the devices and generating the packets that should be sent back to the devices. There are three classes of LoRa end-devices (nodes), which differ only with regards to the down-link scheduling which is based on a cellular-like architecture where several base stations hosting the packet forwarder provide point-to-point link to end-devices on the field. Three different node “flavors” exist, where Class A is the basic one, Class B uses Beacon messages for time synchronization and Class C allows for continuous node listening. (ResearchGate 2018.)

LoRa Security - Both signing and encryption are provided by the LoRaWAN protocol for parts of the LoRaWAN packets and are performed using symmetric keys that both to the Node and to the Network Server knows and possibly also known to Application Servers located behind the Network server depending on requirements. The keys are

shared in a way that is based on how a node joins the network. The AES128-bit data encryption algorithm is used to encrypt data. The MAC Payload section of messages is signed to hinder the manipulation of the cipher-text, or of other values. (Miller 2016.)

Pro – LoRa is a much better choice for devices or sensor nodes transmitting every 10 or 15 minutes in networks with a low or medium number of nodes. LoRa is also the very good option for very wide networks, having long-range links. Other communication modules cannot get more than a few km. **Cons** – LoRa is not very good for projects which require high data-rate and/or very frequent transmissions (like every 10 seconds) and LoRa is probably not suitable for highly populated networks. But this depends on the number of nodes as well as on the number of packets per hour that each node sends. LoRa node should be powered by a solar panel, or better, connected to mains electricity as power consumption is a major challenge. Lastly, note that due to the low bandwidth, LoRa by itself does not support Over the Air Programming (OTA), but can be done using 3G, GPRS or WiFi modules that allow OTA as a second radio for OTA purposes. (Libelium Communication Distribution 2018.)

2.2.7. Comparing the Wireless Communication Protocols

The analysis of some few papers and several online articles led to the findings from available publication for different estimations and evaluation results regarding the specifications and performance of wireless protocols. This is because these specifications are gotten from the implementation performed by the producers of these devices and standards. As a result, some main wireless modules from well-known manufacturers are illustrated based on the wireless protocols used in this thesis. Table 4 summarizes some of the main features of the wireless protocols taken from the respective datasheets of each wireless protocol devices used. The references to the datasheets and networking guide of these wireless protocols are given at the reference section of this document.

Table 4. Comparing BLE, XBee, WIFI and LoRa Wireless Protocols.

Technical Specification	BLE	Zigbee (IEEE 802.15.4)	WiFi	LoRa
Device Family	Bluetooth v4.0 /Bluetooth Smart Chipset: BLE112	XBee-PRO 802.15.4 EU	WiFi PRO module	Semtech SX1272 Module
Frequency bands	2400–2500 MHz	ISM 2.4 GHz	2.4GHz IEEE 802.11 b/g/a	863-870 MHz (Europe) 902-928 MHz (US)
Transmission Power	[-23 dBm, +3 dBm]	+10 dBm	802.11b: 17 dBm 802.11g: 14 dBm 802.11a: 12 dBm	+14dBm
RX sensitivity	-103 dBm	-100 dBm	802.11b@11Mbps PER<8%: -87 dBm 802.11g@54Mbps PER<10%: -73 dBm 802.11a MCS0 PER<10%: -86 dBm	-134 dBm
Transmission Range (at maximum TX power)	100 m	750 m	<300m	LOS = 22km (13.4miles) NLOS = +2km (1.2miles)

Maximum over the air data rate	1Mbps	250 Kbps	Max 72.2Mbps (IEEE 802.11n HT)	Not mentioned
Tx current @3.3 VDC	36 mA	215mA	350 mA	Not mentioned
Rx current @3.3 VDC	8 mA	55mA	130 mA	Not mentioned
Encryption	AES 128	AES 128	AES-128/256, 3DES, SSL3/TLS1, HTTPS, RSA, WEP, WPA and WPA2	AES 128/192/256
Authentication	Not mentioned	Not mentioned	WPA-TKIP 128-bit WPA2 CCMP (AES)	Not mentioned
Topology	Scatternet	Star, tree, mesh	Star	Star

2.2.8. Choosing a Wireless Protocol

In choosing a wireless solution we need to consider several points and provide answers to relevant questions that arises to ensure that the wireless communication link will perform satisfactorily. Such questions are as follows; Is it possible to get a clear line-of-sight propagation? Else, can we overcome attenuation and multipathing to provide reliable communication? Do we have an ideal and acceptable location to mount the antenna and equipment? What is the best frequency range for the application? Have the client provided enough information and support to aid with getting the answers to these questions? Since we want to develop an industrial wireless link and we need to consider the distance, reliability and configurability, we are going to make use of a proprietary RF system. (Conley 2018.)

Wireless connectivity offers multiple advantages such as easier installation and maintenance, better flexibility and scalability and a long communication range, and not having to worry about the wires wearing or getting tangled together. However, when selecting

any specific wireless solution, we need to perform a site assessment. It is important to perform some analysis on the communication environment. A site assessment is an analysis of the distance, terrain, obstacles, foliage, potential RF Interference sources and other factors that can affect the optimum operation of the communications link. The site assessment done is based on the challenges inherent in the application especially for more complex or critical applications. For an ideal situation where there is a clear line-of-sight between the transmitter and receiver antennas, there can be good assurance that a wireless link will operate successfully given adequate power in the appropriate frequency range. Still, it is required to put into consideration the probability of the environmental conditions changing seasonally, or other changes in the industrial area and carryout proper investigation on the presence of sources of RF interference nearby. (Conley 2018.)

Wireless protocols are also prone to malicious attackers which might attempt to spy and hack into the network to control or interfere with and jam communications. Therefore, careful considerations and field testing is needed to test if a wireless solution can deliver the required robustness, reliability and security compared to the wired solution.

2.2.9. Basic Network Attacks

The network security has become an important topic to note due to the frequency and variety of existing attacks along with the potential threat of new and more destructive future attacks. Attackers make different types of network attacks based on their interest as some may not only be interested in exploiting software applications, but also want to get unauthorized access of the network and the devices connected to the network. Some types of network attacks are eavesdropping, Data Modification, Identity Spoofing (IP Address Spoofing), Password-Based Attacks, Denial-of-Service (DoS)Attack, Man-in-the-Middle Attack, Compromised-Key Attack, Sniffer Attack and Application-Layer Attack etc. (VSkills 2018.)

These attacks may be classified into passive monitoring of communications, active network attacks, close-in attacks, exploitation by insiders, and attacks through the service

provider, Distributed Attack (Distributed DoS) and Hijack attack etc. Any of these attacks can be used to cause damage to the network and gain unauthorized access of the network and the devices connected to the network. The attacker may be able to control the devices or make unwanted modification to it and its data. (VSkills 2018.)

Security measures should be taken to protect the data and ensure reliability and authenticity of data. Based on the IEEE 802.16e standard, the security measure used should provide strong support for authentication, key management, encryption and decryption, control and management of data protection and security protocol optimization. (VSkills 2018.)

Figure 16 shows some type of network attacks.

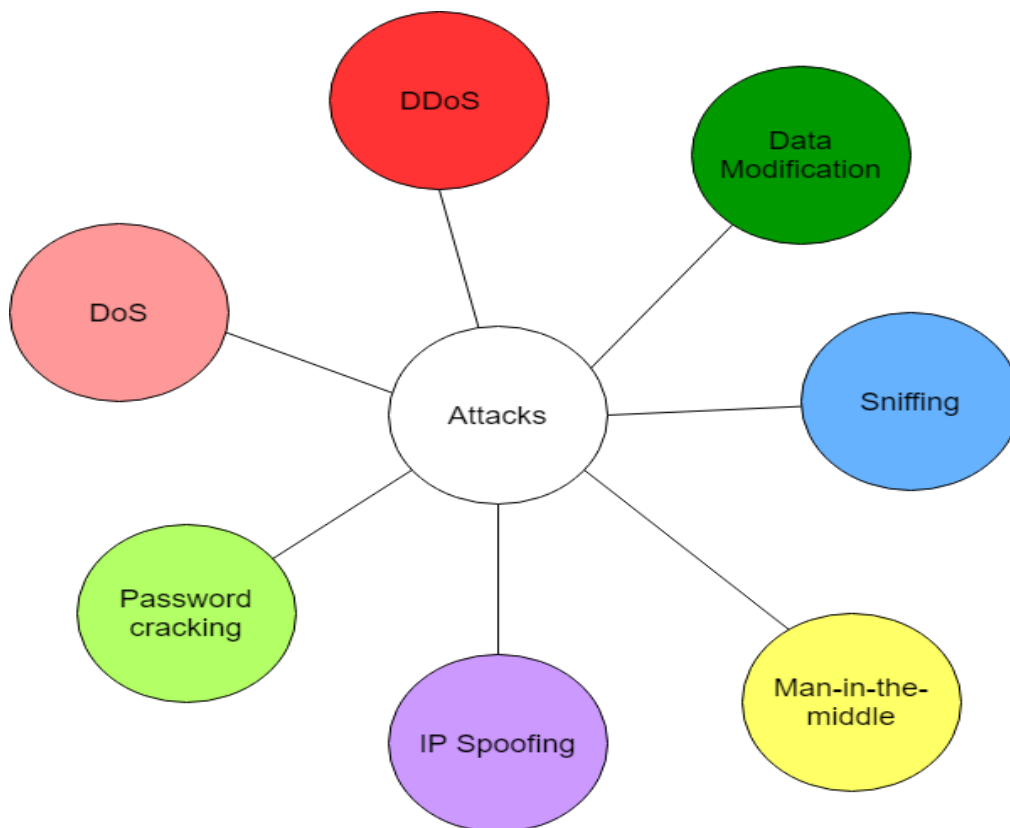


Figure 16. Types of network attacks (PCtech24 2017).

2.2.10. Encryption and Authentication

Wireless communication links certainly comes with an intrinsic vulnerability to security risks and therefore the right steps should be applied to mitigate them. Therefore, we need to make use of wireless systems that has a trusted and accepted security features and capability. Wireless communication links should be able to provide security for data transmission, for example using AES with 128- or 256-bit encryption. In over-the-air transition of data, the US government for example has adopted the AES encryption as the required standard for the secure data transition (Conley 2018).

Therefore, AES encryption can be applied to the smart NO_x CAN data to ensure the secure transmission of the data.

2.3. Smart NO_x Sensor, Speedgoat and Engine Control Module (ECM)

In this section, we discuss the components used such as smart NO_x sensor, speedgoat and the engine control module (ECM).

2.3.1. Smart NO_x Sensor

The smart NO_x is a sensor that measures the oxygen (O₂) percentage and nitrogen oxides (NO_x) ppm in the exhaust of combustion engines. Oxygen is measured as a percentage while NO_x concentration is measured in ppm. (Ina & Bertrand 2010.)

Nitrogen Oxides (NO_x) are a group of poisonous, highly reactive gases of which two occur naturally namely nitric oxide (NO) and nitrogen dioxide (NO₂). The combustion of fossil fuels is the most common source of NO_x emissions. The amount of emission depends on the air-fuel mix ratio as well as the amount of nitrogen in the fuel. At high temperatures and conditions that encourage oxidation NO_x formation in combustion is favored. NO₂ has adverse effects on human health and at high concentrations it can lead to the inflammation of the airways. NO₂ is also responsible for the formation of second-

ary particulate aerosols and ozone (smog (O_3)) in the atmosphere. These are noticeable air pollutants because of their severe impacts on human health. (European Environment Agency (EEA) 2018.)

NO_x can also lead to acid rain and eutrophication. Eutrophication leads to the occurrence of potential changes in the quality of soil and water. This leads to devastating effects on the aquatic ecosystems in rivers and lakes and causes damage to forests, crops and other vegetation. Eutrophication can also bring about decreased biodiversity, changes in species composition and dominance, and toxicity effects. NO_x therefore has both directly and indirectly effects on human health. Sources NO_x include automobiles, trucks and various non-road vehicles such as construction equipment, boats, etc. Other sources are industrial sources such as power plants, industrial boilers, cement kilns, and turbines. Stationary sources of NO_x were required to install and operate reasonably available control technology (RACT) by May 31, 1995 according to the Clean Air Act Amendments of 1990 for the United States. (United States Environmental Protection Agency (EPA) 2018.)

Similarly, according to the Department of Communications, Climate Action and Environment, the EU Clean Air Policy has an interim objective to reduce health and environmental impact up to 2030, these objectives include avoiding 58,000 premature deaths, saving 123,000km² of ecosystems, (including 56,000km² protected Natura 2000 sites) from nitrogen pollution, and saving 19,000km² forest ecosystems from acidification.

The commercial NO_x sensors applied in automotive are basically zirconia (YSZ) electrochemical sensors of the amperometric type. The NO_x sensor's fundamental principle of operation is illustrated in figure 17. The sensor makes use of two or three electrochemical cells in adjacent chambers. The first cell electrochemically pumps O_2 out of the sample to avoid the O_2 interfering with the NO_x measurement in the second cell. The removal of the O_2 makes this type of NO_x sensor to have a dual function; it can also be used in detecting of the O_2 level in the exhaust. (Carstens & Majewski 2018.)

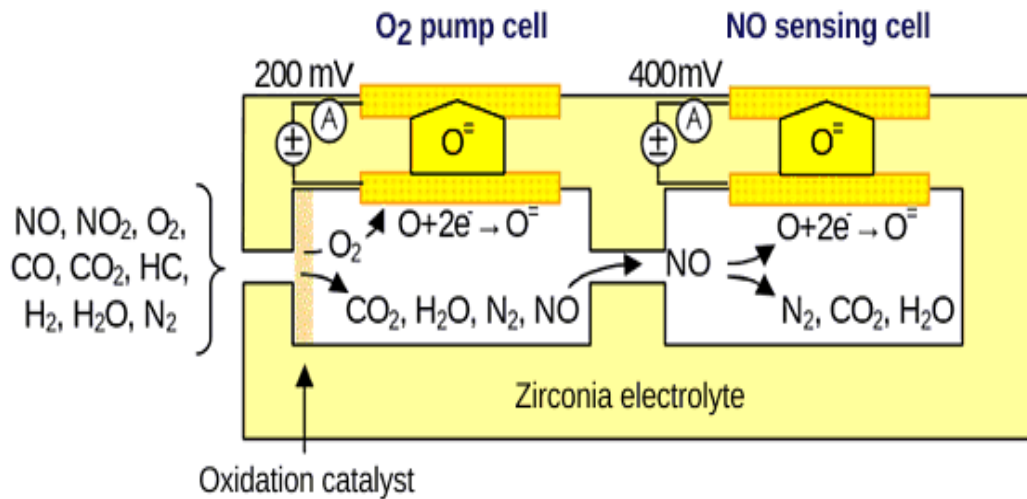


Figure 17. Schematic representation of an amperometric NO_x sensor.

(Carstens & Majewski 2018.)

NO_x sensors comprise of a minimum of two oxygen pump cells (see figure 17) - one removes excess oxygen from the exhaust gas, and the other measures the resultant oxygen concentration from the decomposition of NO_x. When the O₂ in the first cell is reduced, it produces O⁻ ions which are pumped through the zirconia electrolyte by the application of a bias of approximately -200mV to -400mV. The O₂ concentration is determined from the pumping current because it is proportional to the pumping current. The second cell collects the remaining gases where the NO_x decomposes into N₂ and O₂ using a reducing catalyst. Like the first cell, a bias of -400 mV is applied to the electrode to separate the O₂ produces and then pumps out the O₂ from the cell; the second cell's pumping current is proportional to the amount of oxygen from the NO_x decomposition. To help control the NO_x sensing cell, an additional electrochemical cell can be applied as a Nernstian lambda sensor. To avoid interference, all HC and CO in the exhaust gas is oxidized before reaching the NO_x sensing cell and any NO₂ in the sample is converted to NO before the NO_x sensing begins to guarantee that the sensor output is proportional to the NO_x concentration. (Carstens & Majewski 2018.)

The NO_x sensor has been applied recently in urea-SCR (Selective Catalytic Reduction) systems for light- and heavy-duty diesel engines. SCR systems basically makes use of a NO_x sensor downstream of the SCR catalyst to satisfy various OBD (on-board diagnos-

tics) requirements. Excessive NO_x or ammonia concentrations in the SCR outlet leads to an OBD malfunction notification because NO_x sensors are sensitive to NO_x and ammonia gases. The SCR is a system that injects a solution like AdBlue through a catalyst in the exhaust to react with the nitrogen oxide gas produced by the combustion process. AdBlue is a solution made up of urea and water injected into the engine/vehicle's exhaust system to breakdown the harmful nitrogen oxide into harmless nitrogen and oxygen gases before it comes out of the exhaust pipe. However, note that the NO_x sensor measures the NO_x concentration before the NO_x is reduced or broken down. (Parkers 2018.)

In the Wärtsilä's smart NO_x sensor case, the current installation has the smart NO_x sensor connected to the engine control unit (ECU) with a wired CAN bus connection. The smart NO_x sensor data is transmitted using the SAE J1939 protocol which is built on top of CAN Networks. SAE J1939 was developed specifically for use in heavy duty environments, with the aim on achieving reliable and fault tolerant communications. The objective of the test case in this thesis is to investigate and simulate the possibility of replacing the existing wired connection between the smart NO_x sensor and the rapid control prototyping system (speedgoat), and possibly in the future the Engine Control Unit (ECU) with a wireless communication solution.

2.3.2. Acquiring data from the Smart NO_x sensor

According to the datasheet of the smart NO_x sensor provided by Wärtsilä, SAE J1939 is used, with extended 29-bit CAN frame identifiers and a transfer rate of "250kBaude". The smart NO_x sensor transmits data using the address "18F00F52h" when pin5 is open. A new CAN frame is transmitted every 50ms. Table 5 illustrates the format of data bytes in each transmitted CAN frame. Further details about status bytes are available in the datasheet. (Ina & Bertrand 2010.)

To obtain correct readings, the sensor needs to be heated first. Heating must be initiated externally by sending the 8 bytes hexadecimal heating signal "04h" at a repetition rate > 100ms.

Table 5. Payload in smart NOx CAN frames.

	7	6	5	4	3	2	1	0
0 (L-Byte)	NOx	NOx	NOx	NOx	NOx	NOx	NOx	NOx ←
1 (H-Byte)	NOx ←	NOx	NOx	NOx	NOx	NOx	NOx	NOx
2 (L-Byte)	O ₂	O ₂	O ₂	O ₂	O ₂	O ₂	O ₂	O ₂ ←
3 (H-Byte)	O ₂ ←	O ₂	O ₂	O ₂	O ₂	O ₂	O ₂	O ₂
4	Status Byte	Status Byte	Status Byte	Status Byte	Status Byte	Status Byte	Status Byte	Status Byte
5	not used**	Status Heater Mode	Status Heater Mode	Error* Heater	Error* Heater	Error* Heater	Error* Heater	Error* Heater
6	not used**	not used**	not used**	Error* NOx	Error* NOx	Error* NOx	Error* NOx	Error* NOx
7	not used**	not used**	not used**	Error* O ₂	Error* O ₂	Error* O ₂	Error* O ₂	Error* O ₂

2.3.3. Testing the Smart NOx sensor

To test if the obtained sensor had been connected and powered correctly as well as to examine the transmitted CAN frames, a Kvaser Leaf Light HS v2 USB to CAN interface was used. The sensor is powered by a regulated DC power supply at 24V. Sending the 8 bytes hexadecimal heating signal “04h” to the smart NOx sensor with a Receive ID 0x18FEDF00 makes the smart NOx sensor to start heating and then sends back its CAN frames through the CAN Bus to the wireless module for transmission to the receiver module where the Kvaser Leaf Light HS v2 USB to CAN Bus interface is used to view the data. Repeating the 8 bytes hexadecimal heating signal “04h” every 100ms will maintain the heating of the smart NOx. The result is shown in figure 18. (Ina & Bertrand 2010.)

Time	Tx/Rx	Channel	Msg Type	ID	Message	DLC	Data Byte(s)
12:23:33:7767	Rx	1	x	0x18FEDF00	0x18FEDF00	8	00 00 00 00 00 00 00 04

Figure 18. CAN frame to start heating smart NOx.

2.3.4. Calculating O₂% and NO_x ppm

According to the smart NO_x sensor datasheet by Ina & Bertrand 2010, O₂ percentage can be calculated from O₂ bytes in table 5 using equation 12.

$$O_2(\text{percentage}) = 0.000514\% \times O_2 - 12 \quad (12)$$

NO_x ppm can be calculated from NO_x bytes in table 5 using equation 13.

$$NOx(\text{ppm}) = 0.05 \times NOx - 200 \quad (13)$$

2.3.5. Speedgoat and the Engine Control Module (ECM)

This thesis investigates the possibility of replacing the existing wired CAN bus connection between the smart NO_x sensor and the rapid control prototyping system speedgoat and possibly in the future the Engine Control Unit (ECU) with a wireless communication solution. The speedgoat applies Real-time systems with Simulink Real-Time™ from MathWorks to various applications across many industries, in the lab, field, classroom, or embedded in machinery. Speedgoat solutions and simulink are seamlessly integrated and allows for fast test run of simulink software designs with hardware. (Speedgoat GmbH 2007-18.)

The Engine Control Module (ECM) in figure 19 which can also be called Engine Control Unit (ECU) is a kind of electronic control unit that manages the control of series of actuators on an internal combustion engine to ensure that the engine's performance is optimal. This is done by reading the values from all the sensors within the engine bay and interpreting the data using multidimensional performance maps (referred to as lookup tables) and adjusting the engine actuators accordingly. (Wikipedia 2018a.)



Figure 19. Engine Control Unit of a 1996 Chevrolet Beretta (Wikipedia 2018b).

3. SMART NOX AND SPEEDGOAT WIRELESS COMMUNICATION

This thesis is based on the case study of Wartsila’s smart NO_x sensor. Its aimed at investigating the possibility of using a wireless protocol to send the data of the smart NO_x sensor located on diesel engines to the speedgoat/Engine Control Module (ECM).

The project is aimed at being a low powered wireless solution that will be used to transmit data (CAN frames) of the smart NO_x sensor (connected to the wireless transmitter module) to the wireless receiver module. The receiver module will then relay the CAN frames through an external CAN controller to the speedgoat – performance real-time target machine. A matlab simulink module has been programmed into the speedgoat to receive CAN frames, calculate O₂% and NO_x ppm and display the results on a monitor connected to the speedgoat.

Regardless of if you are making the changes to an existing system or equipment or if you are developing a new infrastructure, distance, barriers and interference can be a challenge. Sometimes the use of remote monitoring and control can be expensive, how-

ever, the cost of having hardwired connections can make an application non-feasible as well as non-viable. Also, hardwiring is basically not achievable in some situations. When adding I/O within an existing system, long distances may not be considered, however, the cost and difficulties accompanying the addition of conduit and wiring to an existing system may exceed the cost and flexibility of making use of a wireless communication link. (Conley 2018.)

3.1. System Architecture

There are some factors considered during the implementation of each wireless protocol such as Receiver Signal Strength Indicator (RSSI), packet loss, bit error rate, latency and power consumption. Also, the aim is to have a designed prototype that is cost effective, robust and reliable, therefore, the choice of the components and products used were carefully carried out.

The list of the hardware components used includes smart NO_x sensor, speedgoat, CAN Bus module, Multiprotocol Radio Shield, Arduino development board, Waspote development board, Waspote expansion board, XBee PRO module, XBee Explorer USB, X-CTU tool, LoRa module, WIFI PRO module and BLE module. These components are discussed briefly in this section and the connection and programming of the components are discussed in subsections 3.2.1 to 3.2.11.

The CAN Bus Module used for XBee, LoRa and WIFI protocol implementation is a CAN 2.0B - Extended CAN frame with 29-Bit identifier from Libelium. (See figure 20.) It has a CAN controller MCP2515 and a CAN transceiver MCP2551. The technical details of the CAN Bus are mentioned in table 6.



Figure 20. CAN Bus Module (Cooking-Hacks 2018).

Table 6. Technical details of the CAN Bus Module. (Cooking-Hacks 2018).

CAN Bus	
Standard	ISO 11898
Cabling	Twisted pair
Connector	DB9
Network Topology	Multi-master
Speed	125 to 1000 Kbps
Signaling	Differential
Voltage Levels	0-5V
Signals	Half Duplex

The CAN Bus modules in figure 20 also provides a 120-ohm termination resistor. The schematic is like the schematic of Mikroelektronika CAN SPI click board illustrated in APPENDIX 1. A new CAN Bus API was written for the Libelium CAN Bus module of figure 20 to implement the 29-bits extended ID of the smart NOx sensor.

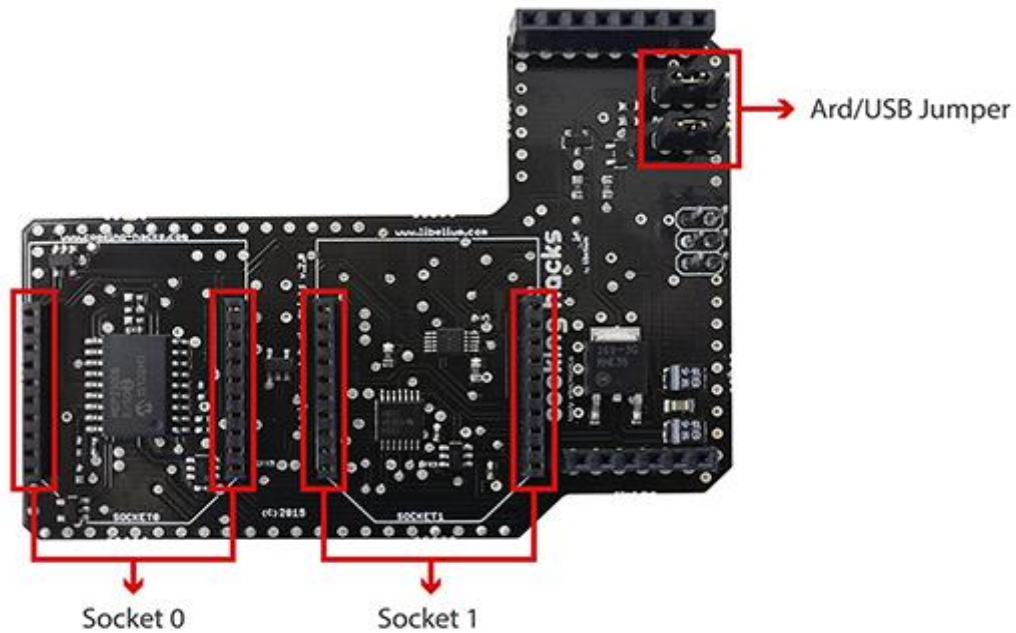


Figure 21. Multiprotocol Radio Shield v2.0 (Cooking-Hacks 2018).

The Multiprotocol Radio Shield can be used as an interconnection shield for Arduino and was designed to allow the connection of two communication modules at the same time. With its SPI bus connections, it can be used to combine any of the following RS-485, CAN Bus, LoRa modules, LoRaWAN, RFID, XBee and Bluetooth. See the Multiprotocol Radio Shield in figure 21. (Cooking-Hacks 2018).

The Arduino development board used is the Arduino UNO Rev3 (see figure 22). It is an open-source microcontroller board developed by Arduino.cc based on the Microchip ATmega328P microcontroller. There are some sets of pins on the board, digital and analog input/output (I/O) pins that can be interfaced with various expansion boards and shields and other circuits for various applications. The Arduino IDE in figure 23 makes use of a version of C++ that has been simplified to make programming it easier. (Sparkfun 2018.)

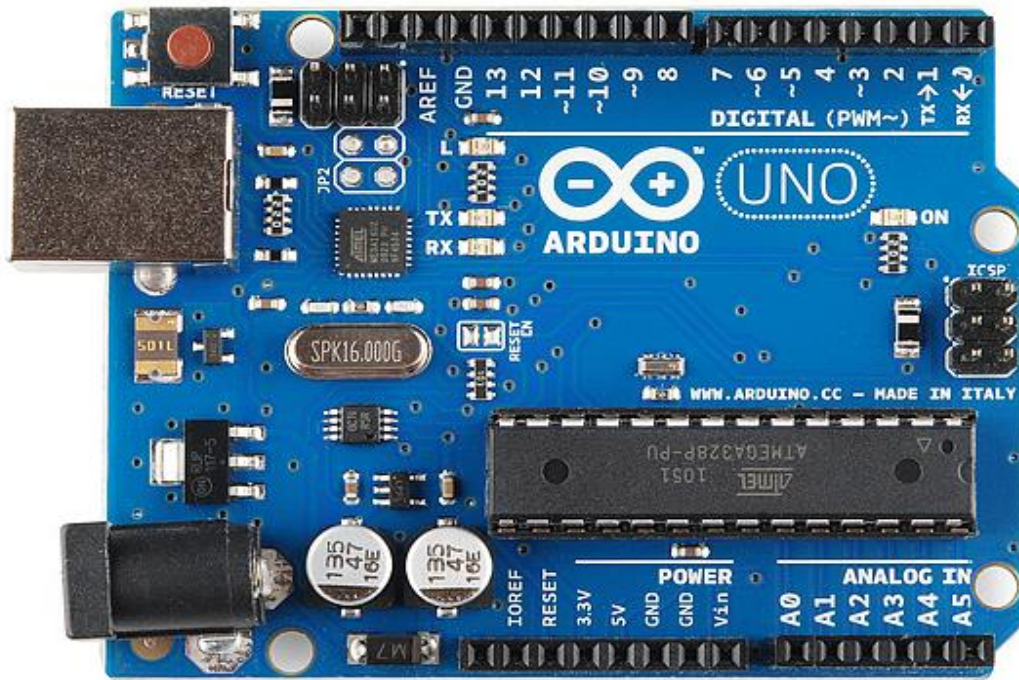


Figure 22. Arduino UNO Rev.3 (Sparkfun 2018).

```

XBee_CAN_Send
#include <multiprotocolShield.h>
#include <Wire.h>
#include <MCP23008.h>
#include <mcp_can.h>
#include <XBee802SendCANData.h>

void setup()
{
  CANconfig();
  XBeeConfigSend();
}

void loop()
{
  SendHeatCANMsg();// send heating data per 100ms
  receiveCANDataSend();// receive and send data per 100ms
  delay(100);
}

```

At the bottom right of the IDE window, it says 'Arduino/Genuino Uno on COM9'.

Figure 23. Arduino IDE.

The structure of the Arduino IDE code is divided into two basic parts namely *setup* and *loop*. They are executed in a sequential order with the *setup* being the first part of the code that is run only once on initialization of the code. This is the part of the code where it is recommended to include the initialization of the modules which are to be used. The *loop* part of the code runs continuously, in an infinite loop. This is where the main part of the code to perform the desired function is included. (Tutorialspoint 2018.)

The Wasmote development board uses the Atmel ATmega1281 microcontroller. The board has some features that improves its performance and application such as the hibernate mode, sleep mode, watchdog and indication LEDs used for several debugging and application purposes. The Wasmote development board is presented in figure 24. (Libelium 2018a.)

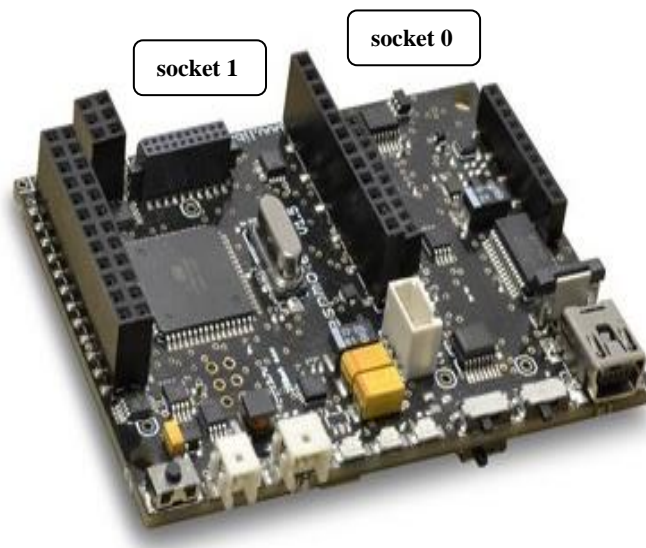
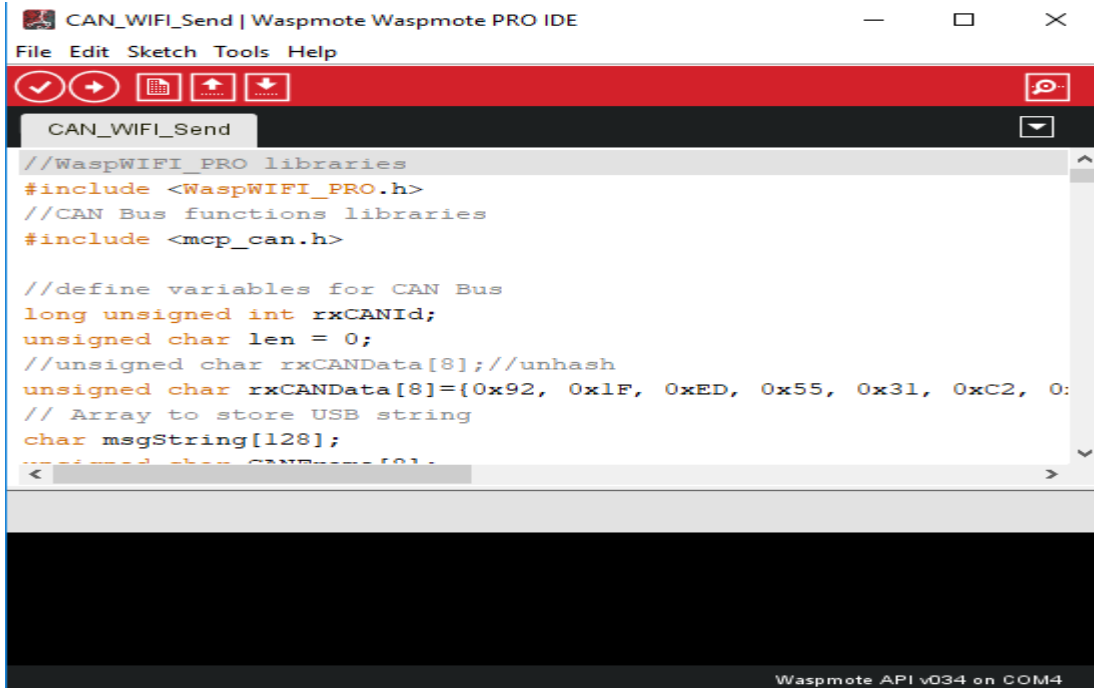


Figure 24. Wasmote development board (Libelium 2018a).

In the Wasmote IDE illustrated in figure 25, the structure of the codes is divided into two basic parts namely *setup* and *loop*. Their function is the same as in the case of the Arduino IDE.



```

CAN_WIFI_Send | Waspnote Waspnote PRO IDE
File Edit Sketch Tools Help
CAN_WIFI_Send
//WaspWiFi_PRO libraries
#include <WaspWiFi_PRO.h>
//CAN Bus functions libraries
#include <mcp_can.h>

//define variables for CAN Bus
long unsigned int rxCANId;
unsigned char len = 0;
//unsigned char rxCANData[8]; //unhash
unsigned char rxCANData[8] = {0x92, 0x1F, 0xED, 0x55, 0x31, 0xC2, 0:
// Array to store USB string
char msgString[128];
unsigned char CANFrame[8];

```

Waspnote API v034 on COM4

Figure 25. Waspnote IDE.

The Waspnote expansion board in figure 26 allows the connection of two communication modules at the same time. This means it can be used to combine any of the following RS-485, CAN Bus, LoRa modules, LoRaWAN, RFID, 802.15.4, ZigBee, DigiMesh, 868 MHz, 900 MHz, LoRa, WiFi, GPRS, 3G, 4G, Sigfox, LoRaWAN, Bluetooth Pro, Bluetooth Low Energy and RFID/NFC which are available for Waspnote. (Libelium 2018b.)

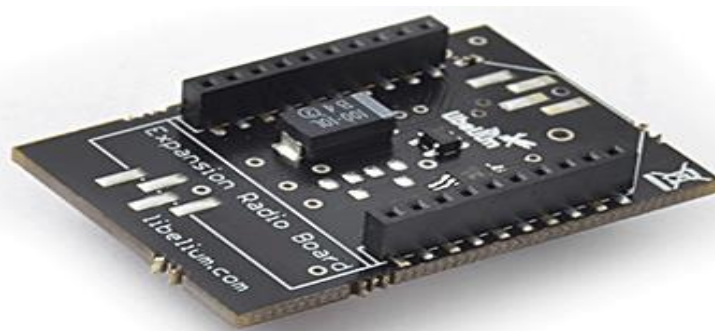


Figure 26. Waspnote Expansion Board (Cooking-Hacks 2018).

The XBee-PRO modules in figure 27 add functionalities such as the node discovery (where specific information is appended to the packet headers so that they can discover other nodes in the same network) and duplicated packet detection to the physical level as well as the link level (MAC layer) already defined by the standard IEEE 802.15.4 which the XBee PRO module complies with. It uses the free frequency band of 2.4 GHz, utilizing 12 channels with a bandwidth of 5 MHz per channel as shown in table 7. (Libelium 2017a.)

Table 7. XBee 802.15.4 Channel Number Frequency. (Libelium 2017a).

Channel Number Frequency	
0x0C – Channel 12	2.405 – 2.410 GHz
0x0D – Channel 13	2.410 – 2.415 GHz
0x0E – Channel 14	2.415 – 2.420 GHz
0x0F – Channel 15	2.420 – 2.425 GHz
0x10 – Channel 16	2.425 – 2.430 GHz
0x11 – Channel 17	2.430 – 2.435 GHz
0x12 – Channel 18	2.435 – 2.440 GHz
0x13 – Channel 19	2.440 – 2.445 GHz
0x14 – Channel 20	2.445 – 2.450 GHz
0x15 – Channel 21	2.450 – 2.455 GHz
0x16 – Channel 22	2.455 – 2.460 GHz
0x17 – Channel 23	2.460 – 2.465 GHz



Figure 27. XBee PRO Module (Cooking-Hacks 2018).

XBee Explorer USB in figure 28 is used with a configuration tool such as XCTU to configure the XBee modules to talk to each other. It is used to hold the XBee module as illustrated in figure 29.



Figure 28. XBee Explorer USB (ES Electronics-Shop 2018).

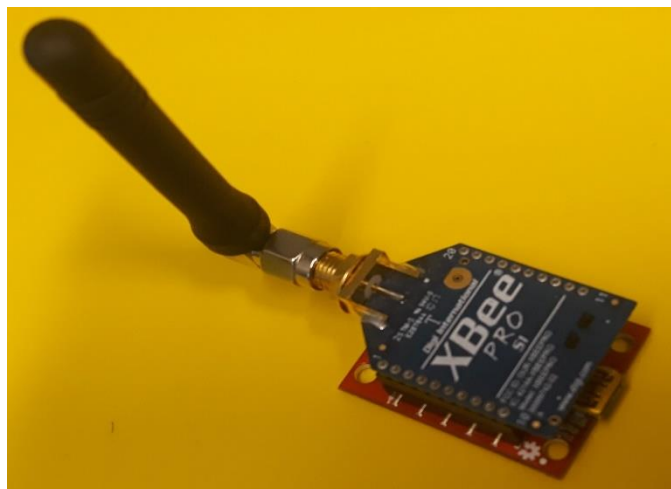


Figure 29. XBee PRO Module on XBee Explorer USB.

The X-CTU tool in figure 30 is a utilities configuration and testing tool. It is used to pre-configure the XBee modules to the same channel and PAD ID. This is done to get the XBees to communicate with each other. Further detail is presented in section 3.1.5.

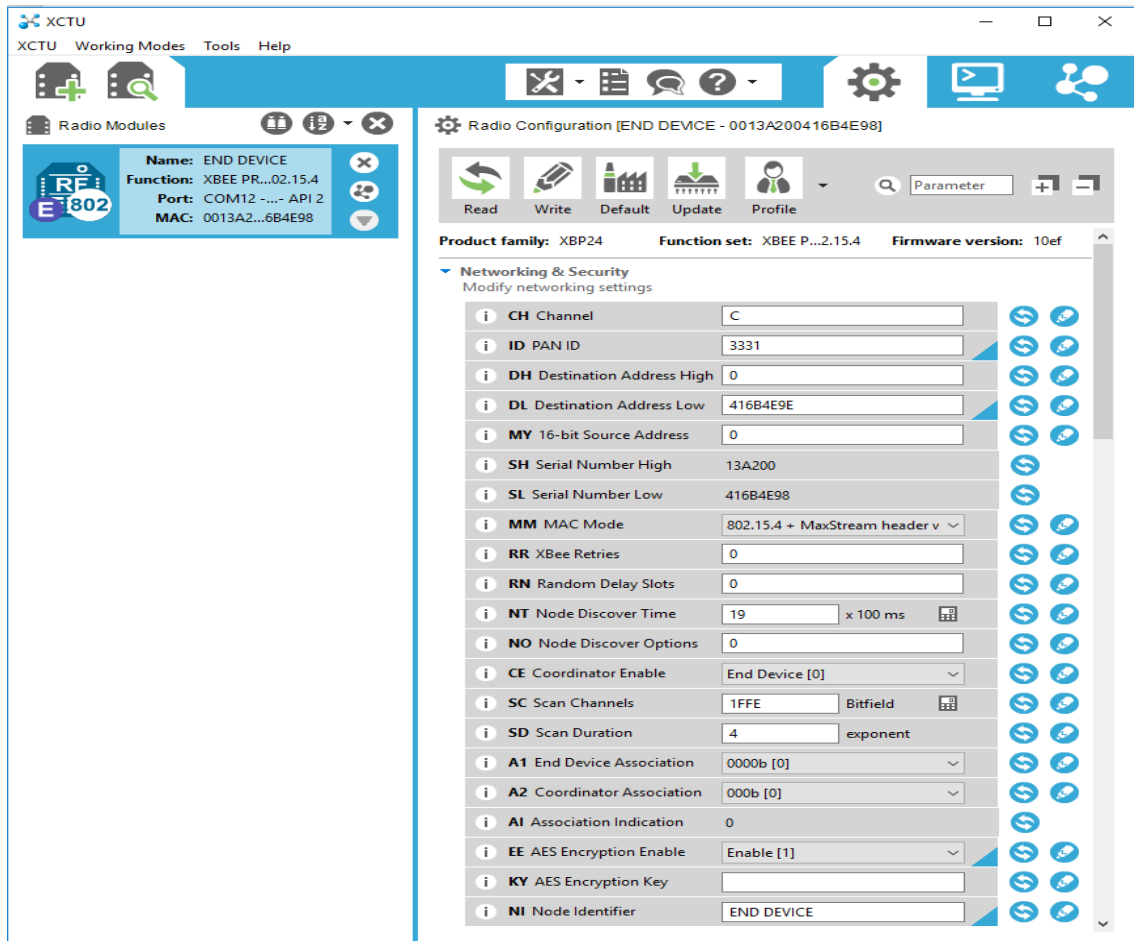


Figure 30. XCTU tool.

The LoRa module in figure 31 provides an optimum range performance due to its receiver sensitivity developed by LoRa™ technology. The module also has a library which enables addressable, reliable and robust communications with ACK, re-tries or time-outs strategies. The frequency can be selected and set with pre-defined channels based on the country in which it is used, that is, it works for both 868 (Europe) and 900 MHz (USA) ISM bands. (Libelium 2017b.)



Figure 31. LoRa Module (Cooking-Hacks 2018).

The LoRa Module specification is presented in table 8.

Table 8. LoRa specification. (Cooking-Hacks 2018.)

LoRa	
Module	SX1272
Dual Frequency Band	863-870 MHz (Europe) and 902-928 MHz (US)
Transmission Power	25 mW
Sensitivity	-134 dBm
Channels	8 (868MHz) and 13 (900MHz)
Range	LOS = 21km (13.4miles) and NLOS = +2km (1.2miles)

The WIFI PRO module shown in figure 32 is an 802.11 b/g radio with 32-bit processor, TCP/IP stack, real-time clock, crypto accelerator, power management unit and analog sensor interface. It is managed by UART and it can be connected to SOCKET0 or SOCKET1 of the Waspote development board. It supports the SSL3/TLS1 protocol used for secure sockets while it supports WEP, WPA and WPA2 WiFi encryption on the WLAN interface. It can connect to any standard router which has been configured as Access Point (AP) and can send data to other devices in the same network as well as send data directly to a web server located on the Internet. (Libelium 2017c.)



Figure 32. WIFI PRO Module.

The WIFI PRO module supports the following features ten simultaneous TCP/UDP sockets, DHCP client/server, DNS client, HTTP client, HTTPS client, FTP client, NTP client, Multiple SSIDs, Roaming mode and OTA feature. (Libelium 2017c.)

The BLE modules in figure 33 is a short-range wireless protocol which utilizes the 2.4 GHz band (2402 – 2480 MHz) and it comprises of 37 data channels and 3 advertisement channels with 2MHz spacing between the channels and GFSK modulation. Although it differs from Bluetooth classic (BR/EDR), it offers similar benefits namely interoperability, robustness and connectivity with smartphones and PCs. UART is used to manage the BLE module and it can be connected on the Wasp mote development board either on SOCKET0 or SOCKET1. It is made to be applied in very low power applications and it conforms with the Bluetooth 4.0 standard, also called Bluetooth Low Energy (BLE). The main features of the module are listed in table 9. (Libelium 2017d.)



Figure 33. Wasp mote Bluetooth Low Energy module (Libelium 2017d).

Table 9. Main features of the BLE module.

BLE Module	
Protocol	Bluetooth v4.0 / Bluetooth Smart
Chipset	BLE112
RX Sensitivity	-103 dBm
TX Power	[-23 dBm, +3 dBm]
Antenna	2 dBi/5 dBi antenna options
Security	AES 128
Range	100 meters (at maximum TX power)
Consumption	sleep (0.4 uA) / RX (8 mA) / TX (36 mA)

The smart NO_x sensor in figure 34 measures the O₂ % and NO_x ppm in the exhaust of combustion engines. The NO_x sensor performance specification according to the Continental smart NO_x sensor datasheet is presented in table 10. (Ina & Bertrand 2010.)

Table 10. NO_x sensor performance specification.

Output Type	Measurement Range	Definition	Data Update Rate
NO _x	-200 – 3012 [ppm] signal: unsigned integer	NO _x -concentration detected by the NO _x -Sensor is transmitted. The transmission is in 0.05 ppm NO _x /bit + 200 ppm. (i.e. 7500 corresponds to 7500 * 0,05 – 200 = 175 ppm)	50ms interval @250 kBaud
O ₂	-12 – 21 [%] signal: unsigned integer	Signal of the actual oxidation factor (% O ₂): The transmission is in 0.000514%/bit + 12%. (i.e. 64202 corresponds to 64202 * 0.000514 – 12 = 21 % O ₂)	



Figure 34. Smart NOx sensor from Wärtsilä.

In this thesis, all the test cases for each wireless solution has the receiver module relaying the CAN frames through an external CAN controller to the speedgoat shown in figure 35. A matlab simulink module has been programmed into the speedgoat to receive CAN frames, calculate $O_2\%$ and NOx ppm and display the results on a monitor connected to the speedgoat.



Figure 35. Speedgoat in VEBIC.

The Matlab Simulink model in figure 36 was created for the speedgoat to handle received CAN frames. The Simulink model is used to continuously poll the client/receiver CAN module 4 times per second, extract data bytes, calculate O₂ % and NO_x ppm and display a continuously updated sliding graph on a monitor connected to the speedgoat.

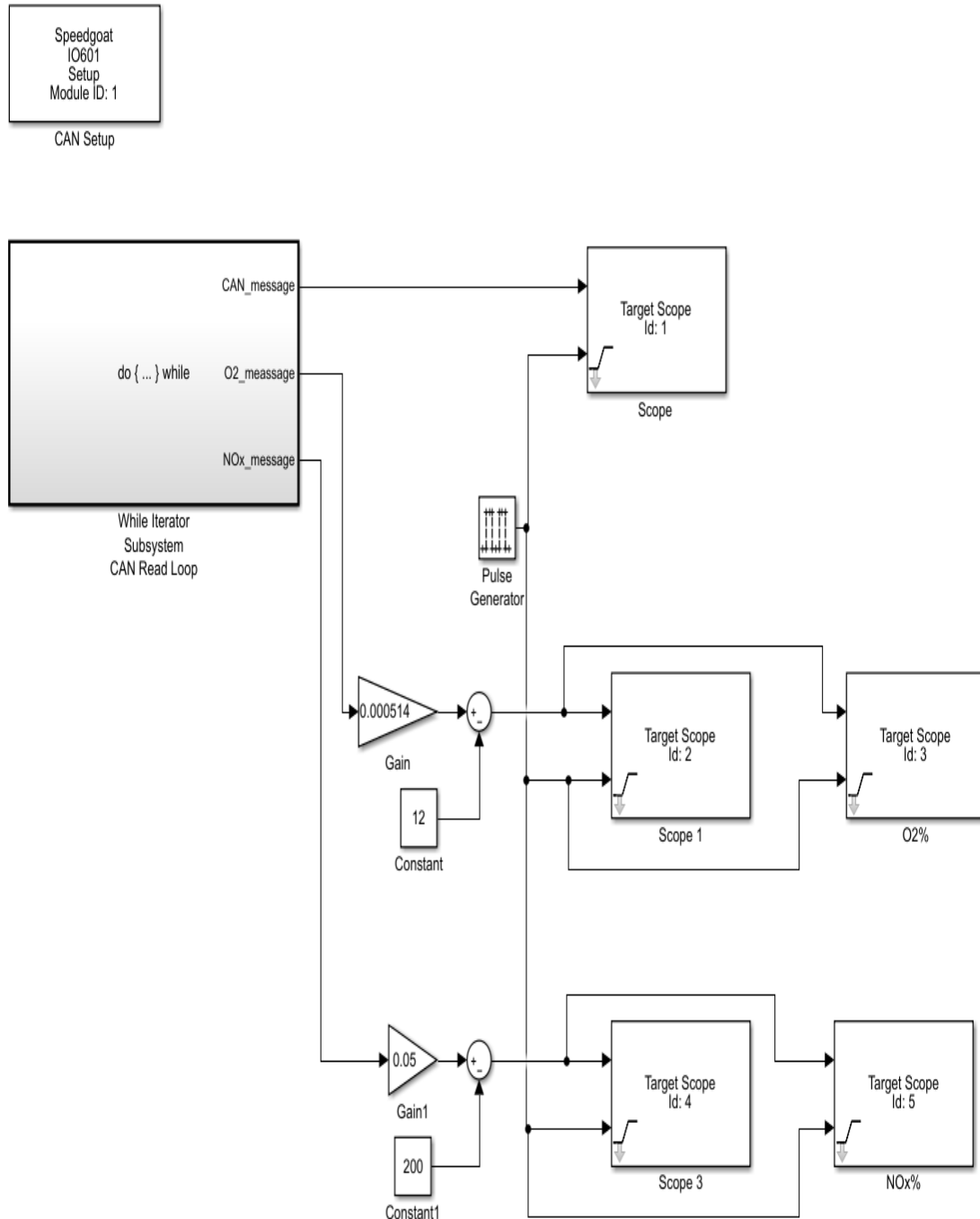


Figure 36. Simulink model to receive smart NO_x CAN frames (Storm 2017).

3.2. System Overview

The system consist of the 24V power supply for the smart NO_x sensor, the smart NO_x sensor is connected to the CAN Bus of the wireless-CAN module (transmitter) and the wireless-CAN modules (receiver) is connected to the speedgoat, the speedgoat has a Matlab Simulink model used to calculate, monitor, and display the O₂ % and NO_x ppm. APPENDIX 2 presents the picture of the Smart NO_x, XBee-CAN Module and Speedgoat system overview.

3.2.1. Connecting the Smart NO_x Sensor

The smart NO_x sensor is connected to the CAN Bus at the transmitter side of the wireless protocol (BLE, XBee, WIFI or LoRa). The CAN High (+) and CAN Low (-) pins of the smart NO_x sensor is connected to the CAN High (+) and CAN Low (-) pins of the CAN Bus respectively. The NO_x 24-volt pin of the smart NO_x sensor is connected to a 24-volt power supply and the NO_x ground pin of the smart NO_x sensor is grounded.

The pin labeling of the smart NO_x sensor is shown in table 11.

Table 11. Smart NO_x sensor pin labeling.

Pin	Description
1	NO _x 24-Volt
2	NO _x Ground
3	CAN Low (-)
4	CAN High (+)

3.2.2. The BLE-CAN bridge Hardware

The hardware components used are the BLE module, CAN Bus, Waspnote PRO extension board and Waspnote PRO development board. A Bluetooth Low Energy communication has been setup with the smart NO_x sensor using a BLE module connected to an

external CAN controller. An Android app “*nRF Connect*” was downloaded and used to test and debug the BLE-CAN implementations. The app can connect to the BLE-CAN server node as a client and subscribe to the notifications of the BLE-CAN server node that is connected to the smart NO_x sensor. It displays readings and status signals sent by the smart NO_x sensor over BLE. The setup in figure 37 provides a proof of concept that can be further developed into a prototype.

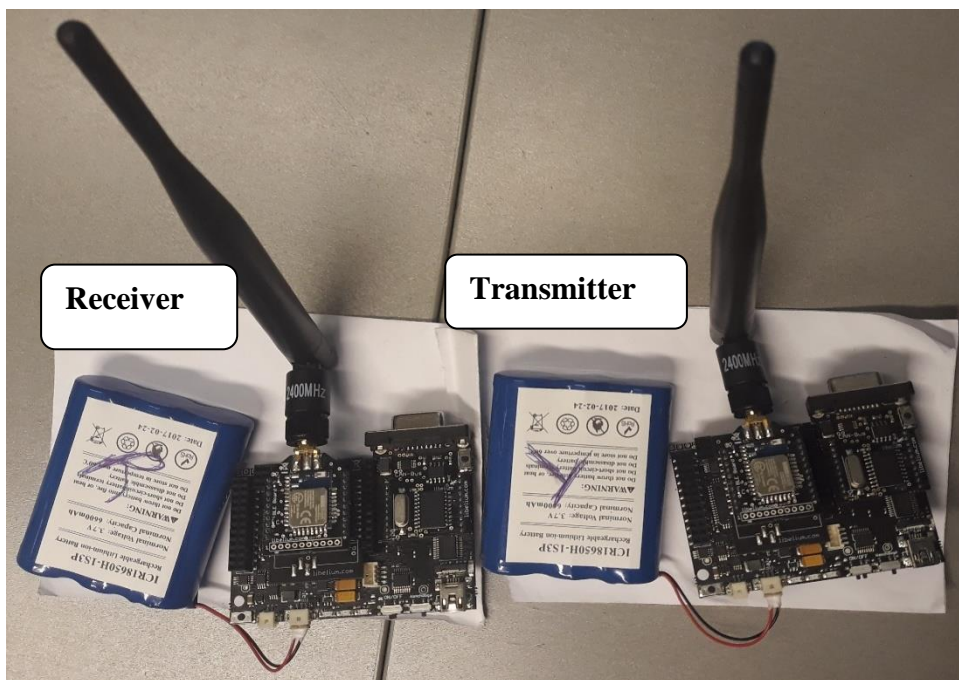


Figure 37. Hardware setup of BLE-CAN bridge.

At the transmitter side, the CAN Bus module is interfaced with the BLE module using the two sockets on the Waspote development board, SOCKET0 and SOCKET1 which make use of UART. The CAN Bus module is used to interface the transmitter BLE module with the smart NO_x sensor. The smart NO_x sensor is connected to the CAN Bus using twisted pair cables (CAN High and CAN Low). At the receiver side, the CAN Bus module is interfaced with the BLE module using the two sockets on the Waspote development board, SOCKET0 and SOCKET1 which make use of UART. The speedgoat is connected to the CAN Bus using twisted pair cables (CAN High and CAN Low).

Figure 38 is a block diagram of the hardware setup of BLE-CAN bridge.

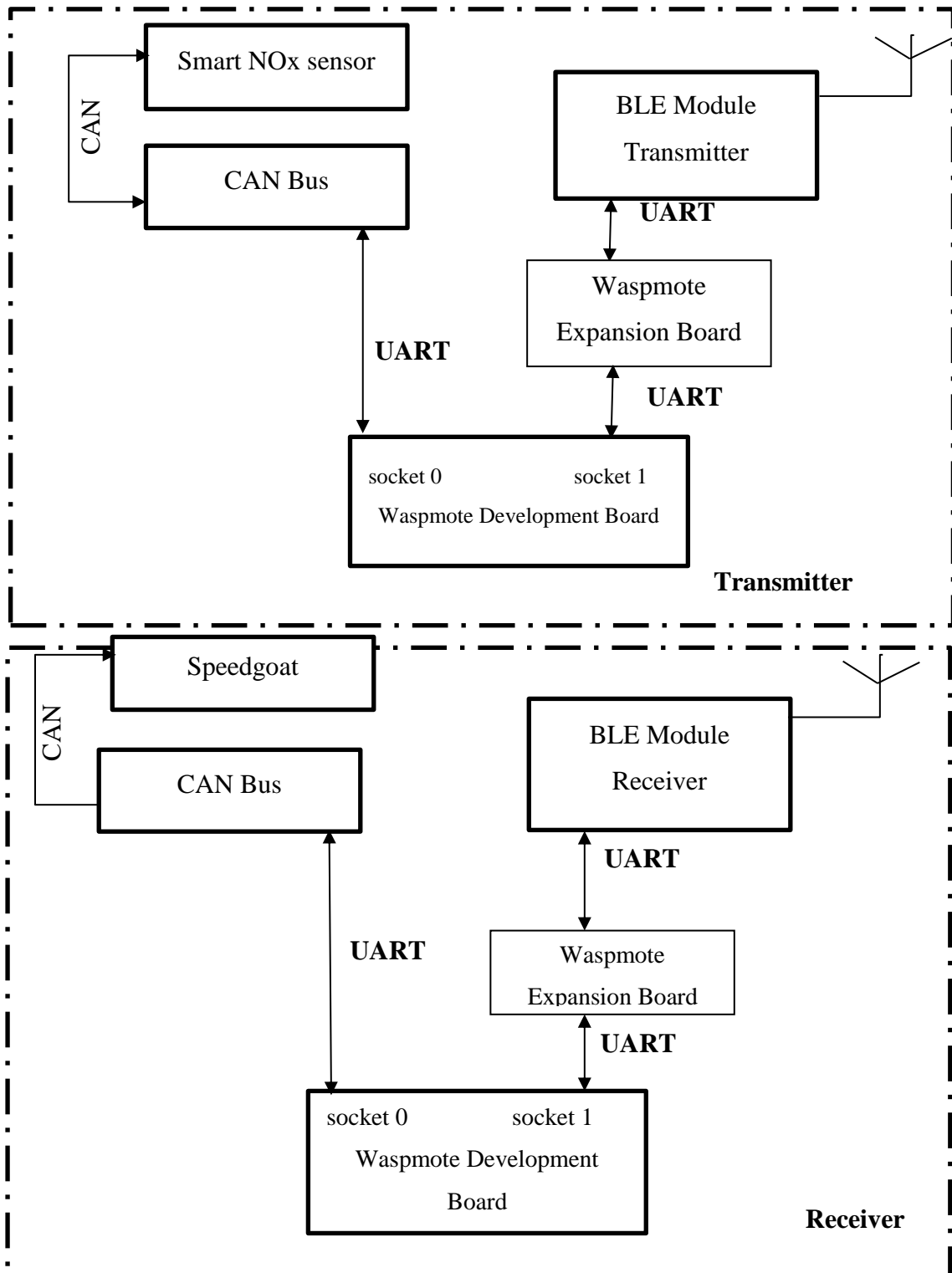


Figure 38. Block diagram for the hardware setup of BLE-CAN bridge.

3.2.3. The BLE-CAN bridge Software

The transmitter and receiver code flowcharts are presented in figure 53 and 54 respectively. The programming of the on the BLE server/transmitter module is such that the CAN Bus was programed to send the 8 bytes hexadecimal heating signal “04h” to the smart NOx sensor to start heating the sensor. The CAN Bus also receives the CAN data sent from the smart NOx sensor after it starts heating and transfers the data through UART to the BLE module for wireless transmission.

The transmitter code has the header file *ConfigBLEServer.h* which is used to configure the BLE Server node to give it a friendly name “BLESenderServer” and initialize the BLE module. The header file *BLESendCANDataAES.h* implements the required C functions to write the start heating command and extract the 8 data bytes from the received CAN frame. It sends the 8 bytes hexadecimal heating signal “04h” to start heating the smart NOx sensor powered by 24V supply. The smart NOx sensor has a 29-bit CAN ID (0x18FEDF00 equivalent in decimal is 419356416) used to send the heat signal from the transmitter side through the CAN Bus to the smart NOx sensor. It also makes the BLE Server node discoverable and connectable, the BLE Server node waits for incoming connections. Once connected, it waits for notification subscribing events and, when they are found, the subscribed attribute is written to allow the master to receive the notification events. It can be programmed to accept incoming connection from only a specific BLE device having the required MAC address (of the BLE Client). The smart NOx sensor and BLE-CAN transmitter setup is presented in figure 39.

The code of the receiver also contains the header file *ConfigBLEClient.h* which is used to configure the BLE Server node to give it a friendly name “BLERecvClient” and initialize the BLE module. The BLE client/receiver module has been programed using the header file *BLERecvCANDataAES.h* to receive the smart NOx sensor data by first looking for the BLE Server node device and then connecting to it. It has been programmed to connect to only a specific MAC address (of the BLE Server node). It then subscribes to notifications of a certain characteristic and wait for notifications from the BLE Server/slave. The BLE client/receiver module receives the data and transfers the data

through UART to the CAN Bus. The CAN Bus is connected to the speedgoat using two twisted pair cable (CAN High and CAN Low). The data is sent to the speedgoat for analysis using the CAN Bus. The Kvaser Leaf Light HS v2 USB can be used to view and debug the CAN data before connection to the speedgoat. The BLE-CAN receiver and Kvaser Leaf Light HS v2 USB setup is shown in figure 40.

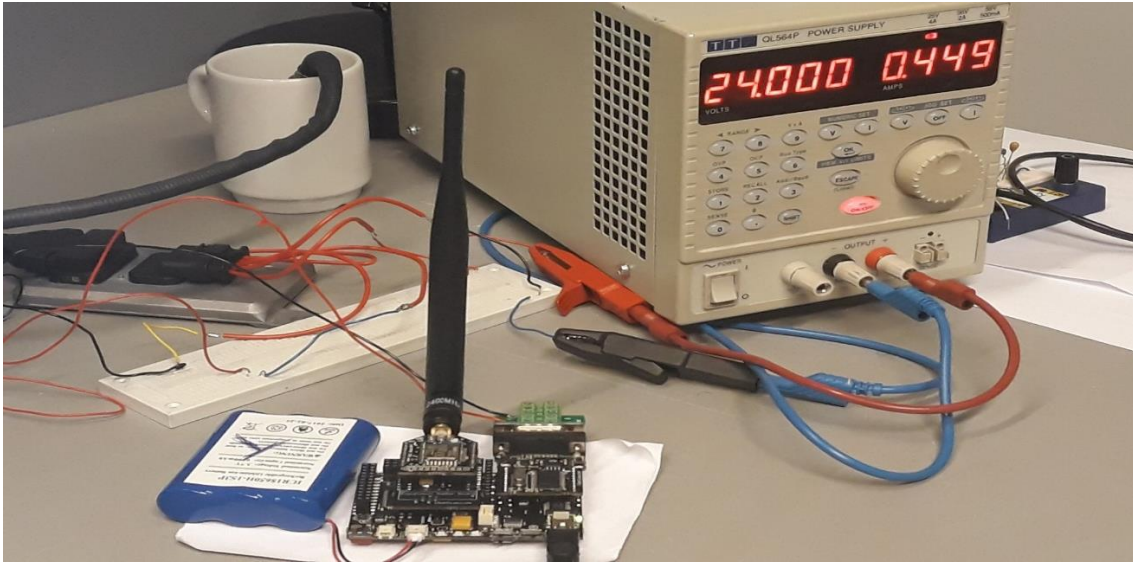


Figure 39. Smart NOx sensor and BLE-CAN transmitter.

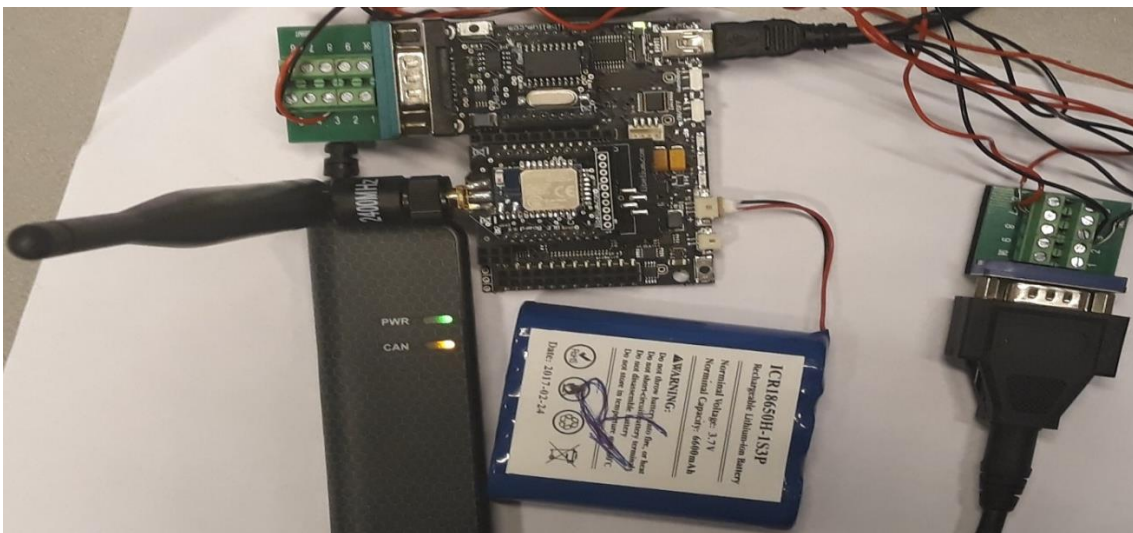


Figure 40. BLE-CAN receiver and Kvaser Leaf Light HS v2 USB.

3.2.4. The XBee-CAN bridge Hardware

The hardware components used are the XBee PRO module, CAN Bus, Multiprotocol Radio Shield and Arduino development board. The XBee-CAN communication has been setup with the smart NOx sensor using a XBee module connected to an external CAN Bus with the help of a multiprotocol radio shield connected over the Arduino Uno rev 3 board.

The code to test and debug the XBee and CAN implementations can send the 8 bytes hexadecimal heating signal “04h” to heat the smart NOx sensor and read signals sent by the smart NOx sensor through the CAN Bus. The data is then sent over XBee PRO module. The setup in figure 41 provides a proof of concept that can be further developed into a prototype.

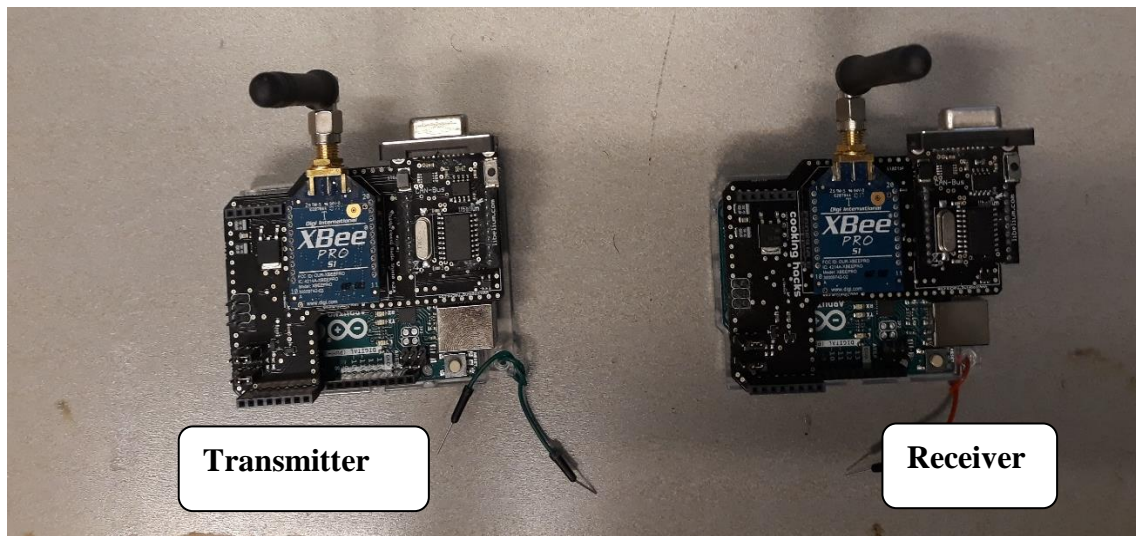


Figure 41. Hardware setup of Xbee-CAN bridge.

At the transmitter side, the Multiprotocol Radio Shield is connected over the Arduino board and the CAN Bus module is placed in socket 0 of the Multiprotocol Radio Shield while the XBee PRO module is placed in socket 1. The CAN Bus module is used to interface the transmitter XBee module with the smart NOx sensor using twisted pair cables (CAN High and CAN Low).

Figure 42 is a block diagram of the hardware setup of XBee -CAN bridge.

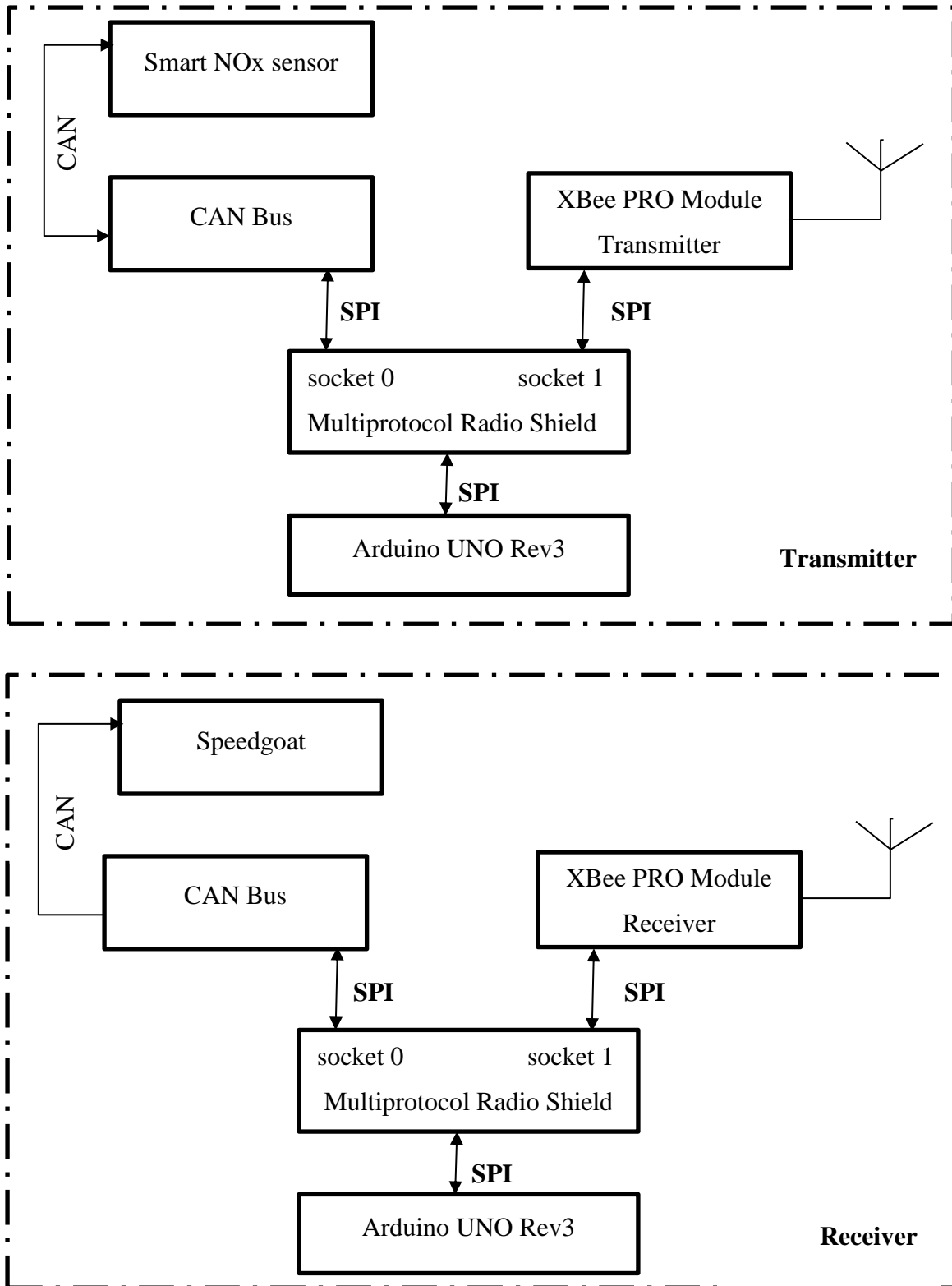


Figure 42. Block diagram for the hardware setup of XBee-CAN bridge.

At the receiver side, the Multiprotocol Radio Shield is connected over the Arduino board and the CAN Bus module is placed in socket 0 of the Multiprotocol Radio Shield while the LoRa module is placed in socket 1. The CAN Bus module was used to interface the receiver XBee PRO module with the speedgoat using twisted pair cables (CAN High and CAN Low).

3.2.5. The XBee-CAN bridge Software

The programming was done on the Arduino IDE environment as seen in APENDIX I. At the transmitter side, the CAN Bus is programed to send the 8 bytes hexadecimal heating signal “04h” to the smart NOx to start heating the sensor to get the data frames from the smart NOx sensor. The CAN Bus also receives the data sent from the smart NOx after it starts heating and transfers the data through SPI to the XBee PRO for wireless transmission. The smart NOx sensor and XBee-CAN transmitter setup is illustrated in figure 43.

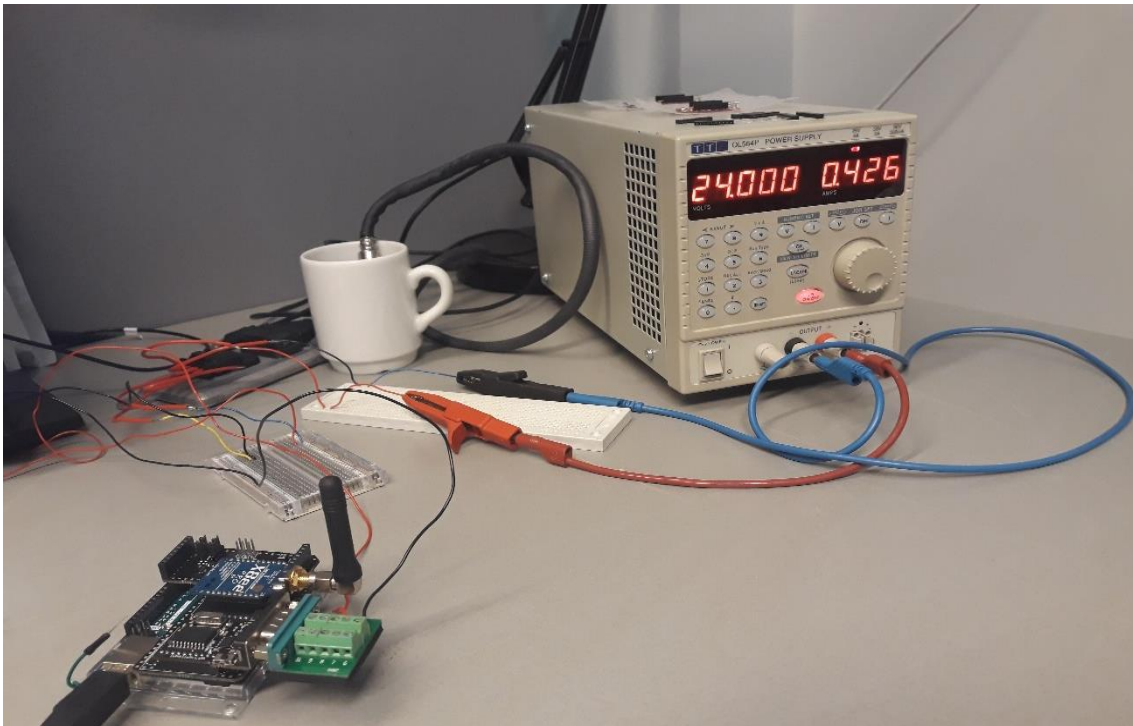


Figure 43. Smart NOx sensor and XBee-CAN transmitter.

At the receiver side, the XBee PRO is programmed to receive the smart NOx sensor data and transfers the data through SPI to the CAN Bus of the receiver module. The CAN Bus is connected to the speedgoat using two twisted pair cable (CAN High and CAN Low) where the received data are analyzed. The Kvaser Leaf Light HS v2 USB can be used to view and debug the CAN data before connection to the speedgoat. The XBee-CAN and Kvaser Leaf Light HS v2 USB setup is shown in figure 44.

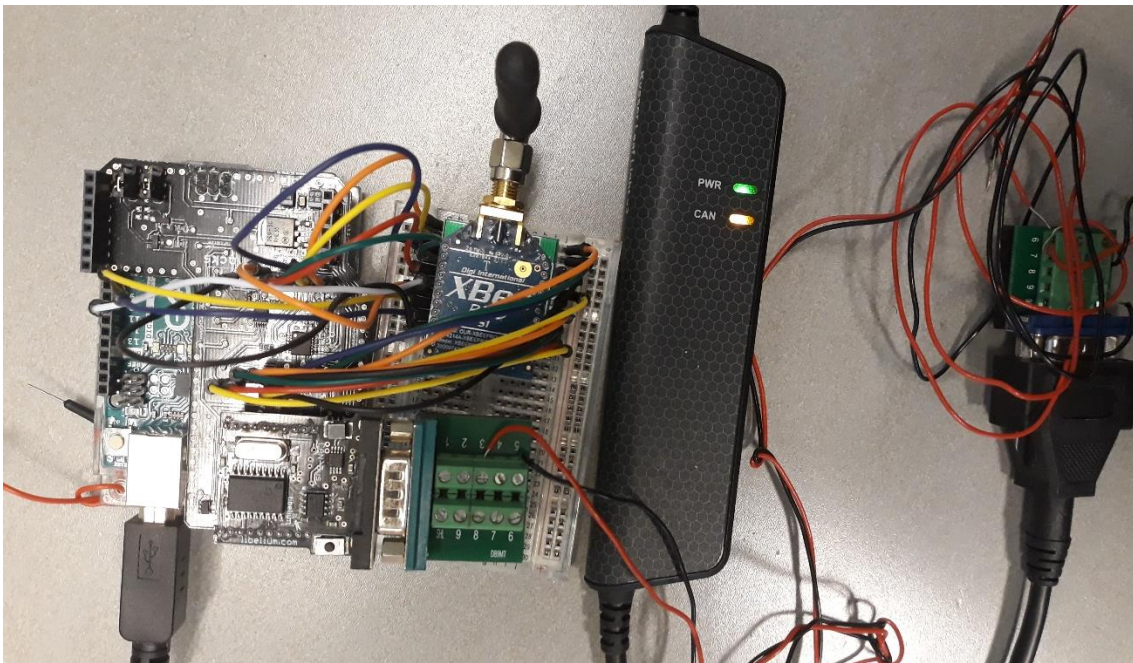


Figure 44. XBee-CAN receiver and Kvaser Leaf Light HS v2 USB.

The flowcharts for the transmitter and receiver codes are illustrated in figure 53 and 54 respectively. The XBee PRO modules are pre-configured using the XCTU software. This is done to get the XBees to communicate with each other. To achieve this, the PAN ID and Channel settings should be the same on each XBee being used. In the case of using only two XBees, the Destination Address High is set to 0 and the Destination Address Low to the Destination Address Low of the other XBee. However, to broadcast to all XBees listening on the same Channel and PAN ID, the Destination High Address is set to 0, and the Destination Address Low to FFFF to enable broadcast mode. For AES Encryption, using the XCTU, enable the “AES Encryption Enable” and provide a key which should be the same for both XBee. Since we are implementing the Xbee using

Arduino, the API Enable should be set to "2", which allows controlling the XBee with Arduino using API commands in the Arduino XBee library.

The code of the transmitter contains the header file *XBee802SendCANDataAES128.h* which is used to initialize the XBee PRO and CAN Bus modules at the transmitter side. It also implements the required C functions to write the start heating command and extract the 8 data bytes from the received CAN frame through SPI interface. It sends the 8 bytes hexadecimal heating signal "04h" to start heating the smart NOx sensor which is powered by 24V supply. The smart NOx sensor has a 29-bit CAN ID (0x18FEDF00 equivalent in decimal is 419356416) used to send the heat signal from the transmitter side through the CAN Bus to the smart NOx sensor. The header file *XBee802SendCANDataAES128.h* is also used read the data from the smart NOx, compute a checksum for error detection, compute a *sender error detection number* (used as a preamble) which is the sum of the received smart NOx data plus checksum (this is compared with the *receiver error detection number* from the receiver module) and implement AES-128 encryption before transmission of the encrypted data using the XBee PRO module. Both the preamble and checksum are appended to the smart NOx data before encryption and transmission.

The code of the receiver contains the header file *XBee802RecieveCANDataAES128.h* used to initialize the XBee PRO and CAN Bus modules at the receiver side. It is also used to receive the CAN frames from the transmitter module, implement AES-128 decryption on the received data, to compute a *receiver error detection number* (this is compared with the *sender error detection number* (preamble) from the transmitter module for error verification). It is the sum of the received data minus the preamble. The header file is also used to transmit the decrypted data to the speedgoat for analysis. The receiver code also has the header files *XBee802PacketLossAES128.h* used for implementing the packet loss measurement and *XBee802RecvRSSIAES128.h* used for RSSI measurement.

3.2.6. The WIFI-CAN bridge Hardware

The hardware components used are the WIFI PRO module, CAN Bus, Wasmote PRO extension board and Wasmote PRO development board. The WIFI -CAN communication has been setup with the smart NOx sensor using a WIFI module connected to an external CAN Bus with the help of Wasmote expansion and development boards. The code to test and debug the WIFI module and CAN module was developed to send the 8 bytes hexadecimal heating signal “04h” to heat the smart NOx and read signals sent by the smart NOx sensor through the CAN Bus. The data is then sent over WIFI. The setup in figure 45 provides a proof of concept that can be further developed into a prototype.

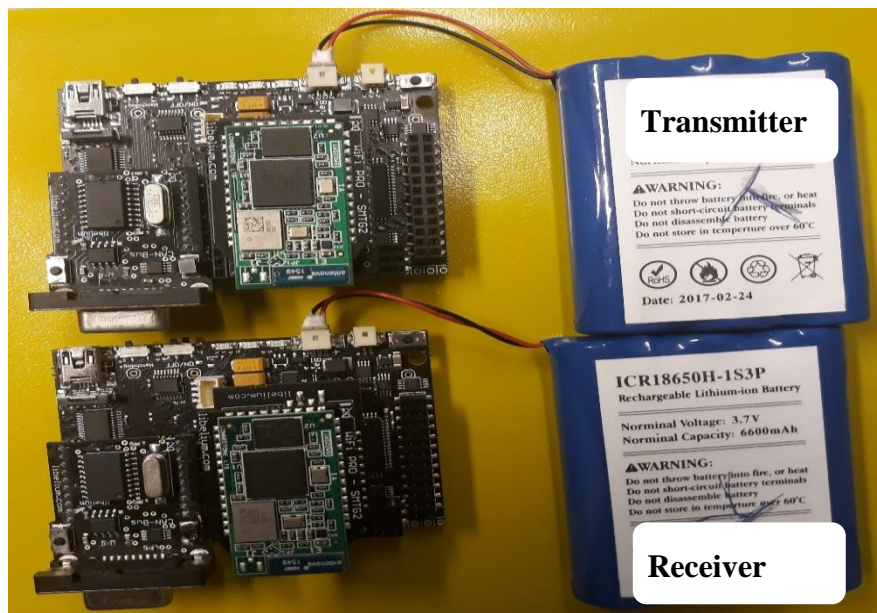


Figure 45. Hardware setup of WIFI-CAN bridge.

At the transmitter side, the Wasmote expansion board is connected on socket 1 of the Wasmote development board and the WIFI module is connected to the Wasmote expansion board. The CAN Bus module is placed on socket 0 of the Wasmote development board. The CAN Bus module is used to interface the transmitter WIFI module with the smart NOx sensor using twisted pair cables (CAN High and CAN Low). At the receiver side, the Wasmote expansion board is connected to socket 1 of the Wasmote development board and the WIFI module is connected to the Wasmote expansion

board. The CAN Bus module is placed in socket 0 of the Waspnote development board. The CAN Bus module is used to interface the receiver WIFI module with the speedgoat using twisted pair cables (CAN High and CAN Low). Figure 46 is a block diagram of the hardware setup of WiFi-CAN bridge.

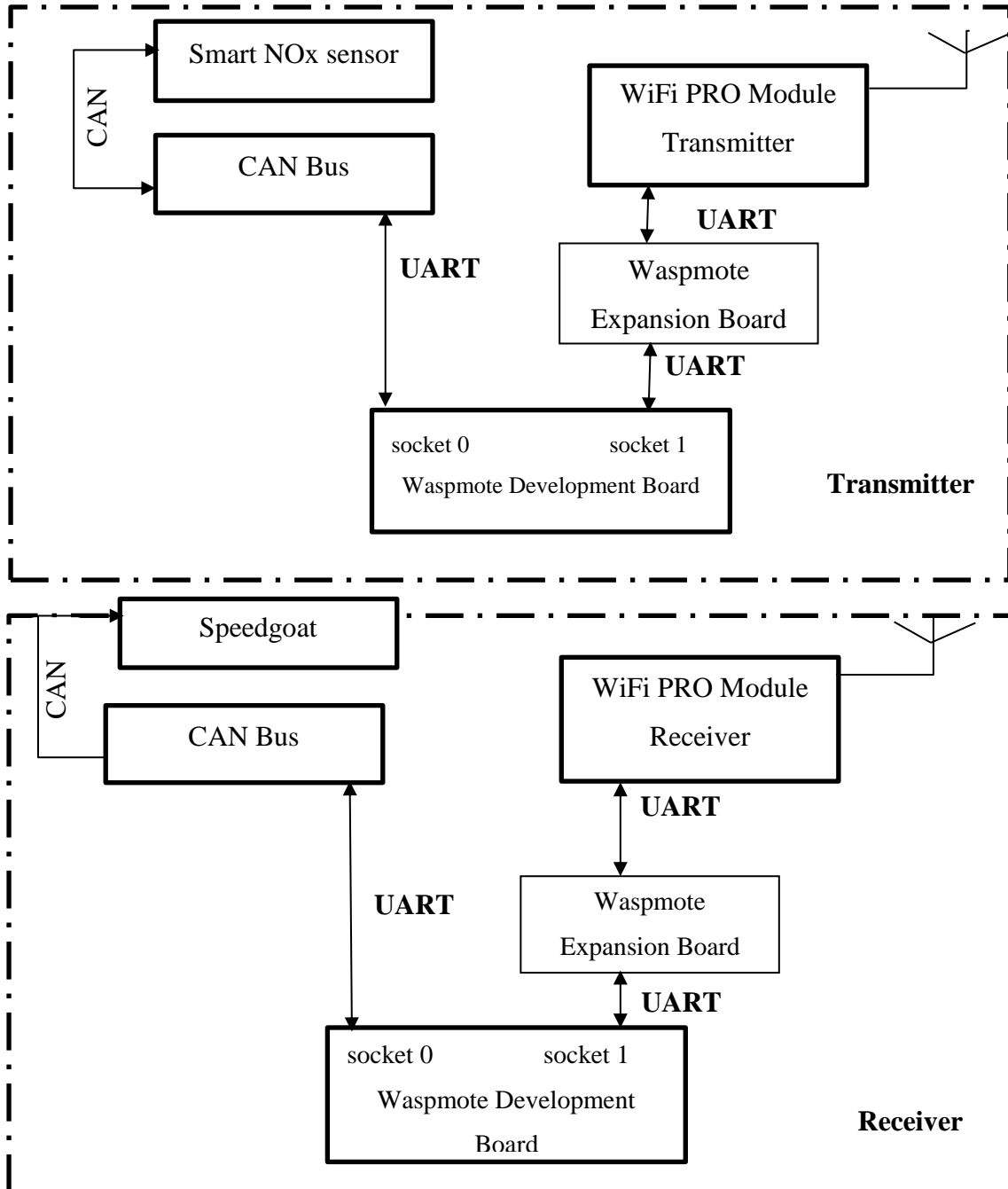


Figure 46. Block diagram for the hardware setup of WIFI-CAN bridge.

3.2.7. The WIFI-CAN bridge Software

The programming was done on the Waspote IDE environment as shown in APENDIX II. At the transmitter side, the CAN Bus was programmed to send the 8 bytes hexadecimal heating signal “04h” to the smart NOx sensor to start Heating the sensor to get the NOx CAN data frames from the smart NOx sensor. The CAN Bus also receives the CAN Data sent from the smart NOx after it starts heating and transfers the data via SPI to the WIFI for wireless transmission. The smart NOx sensor and WIFI-CAN transmitter setup is presented in figure 47.

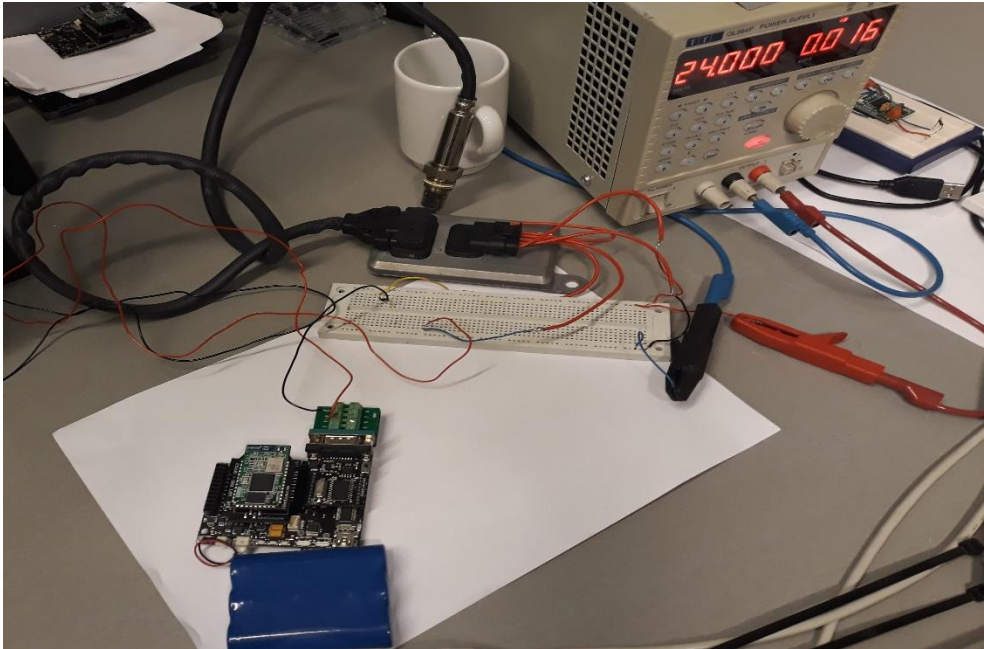


Figure 47. Smart NOx sensor and WIFI-CAN transmitter.

At the receiver side, the WIFI is programmed to receive the smart NOx sensor CAN Data and transfers the data via SPI to the CAN Bus. The CAN Bus is connected to the speedgoat using two twisted pair cable (CAN High and CAN Low) were the received CAN Data are analyzed. The Kvaser Leaf Light HS v2 USB can be used to view and debug the CAN data before connection to the speedgoat. The WIFI-CAN and Kvaser Leaf Light HS v2 USB setup is shown in figure 48.

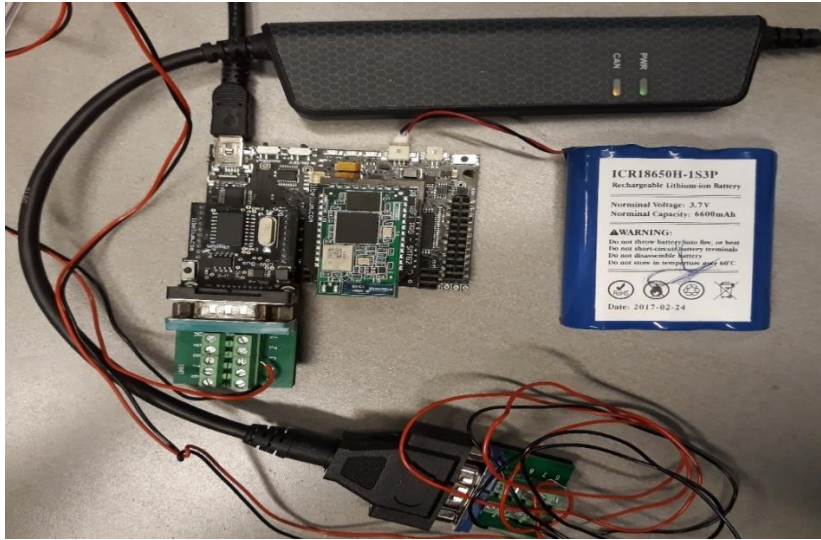


Figure 48. WIFI-CAN receiver and Kvaser Leaf Light HS v2 USB.

The flowcharts for the transmitter and receiver codes are illustrated in figure 53 and 54 respectively. The code of the transmitter contains the header file *ConfigWiFiCANSend.h* which is used to configure the transmitter WiFi module (to join a specific Access Point: ESSID and password must be defined) and to initialize CAN Bus module. The header file *WiFiSendCANDataAES128.h* implements the required C functions to write the start heating command and extract the 8 data bytes from the received CAN frame through SPI interface. It sends the 8 bytes hexadecimal heating signal “04h” to start heating the smart NO_x which is powered by 24V supply. The smart NO_x sensor has a 29-bit CAN ID (0x18FEDF00 equivalent in decimal is 419356416) used to send the heat signal from the transmitter side through the CAN Bus to the smart NO_x sensor. The header file *WiFiSendCANDataAES128.h* is also used read the data from the smart NO_x, compute a checksum for error detection, compute a *sender error detection number* (used as a preamble) which is the sum of the received smart NO_x data plus checksum (this is compared with the *receiver error detection number* from the receiver module) and implement AES-128 encryption before transmission of the encrypted data using the WIFI module. Both the preamble and checksum are appended to the smart NO_x data before encryption and transmission.

The code of the receiver contains the header file *ConfigWiFiCANRecv.h* used to configure the receiver WIFI module (to join a specific Access Point: ESSID and password must be defined) and CAN module at the receiver side. While the header file *WIFIRecvCANDataAES128.h* is used to receive the CAN frames from the transmitter module, implement AES-128 decryption on the received data, to compute a *receiver error detection number* (this is compared with the *sender error detection number* (preamble) from the transmitter module for error verification). It is the sum of the received data minus the preamble. The header file is also used to transmit the decrypted data to the speedboat for analysis. The receiver code also has the header files *WIFIReceivePacketLossAES128.h* used for implementing the packet loss measurement and *WIFIReceiveRSSIAES128.h* used for RSSI measurement.

3.2.8. The LoRa-CAN bridge Hardware

The hardware components used are the LoRa module, CAN Bus, Multiprotocol Radio Shield and Arduino development board. The LoRa-CAN communication has been setup with the smart NOx sensor using a LoRa module connected to an external CAN controller chip with the help of a multiprotocol radio shield connected over Arduino Uno rev 3 board. The code to test and debug the XBee and CAN implementations can send the 8 bytes hexadecimal heating signal “04h” to heat the smart NOx sensor and read signals sent by the smart NOx sensor through the CAN Bus. The data is then sent over the LoRa module. The setup as presented in figure 49 provides a proof of concept that can be further developed into a prototype.

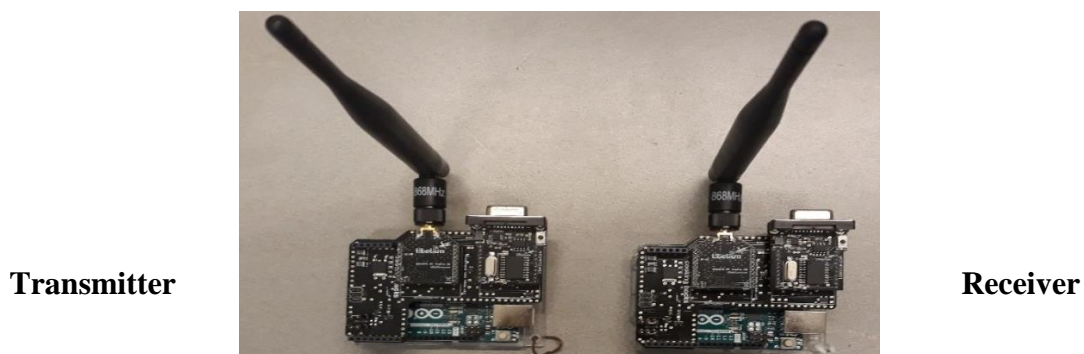


Figure 49. Hardware setup of LoRa-CAN bridge.

Figure 50 is a block diagram of the hardware setup of LoRa-CAN bridge transmitter.

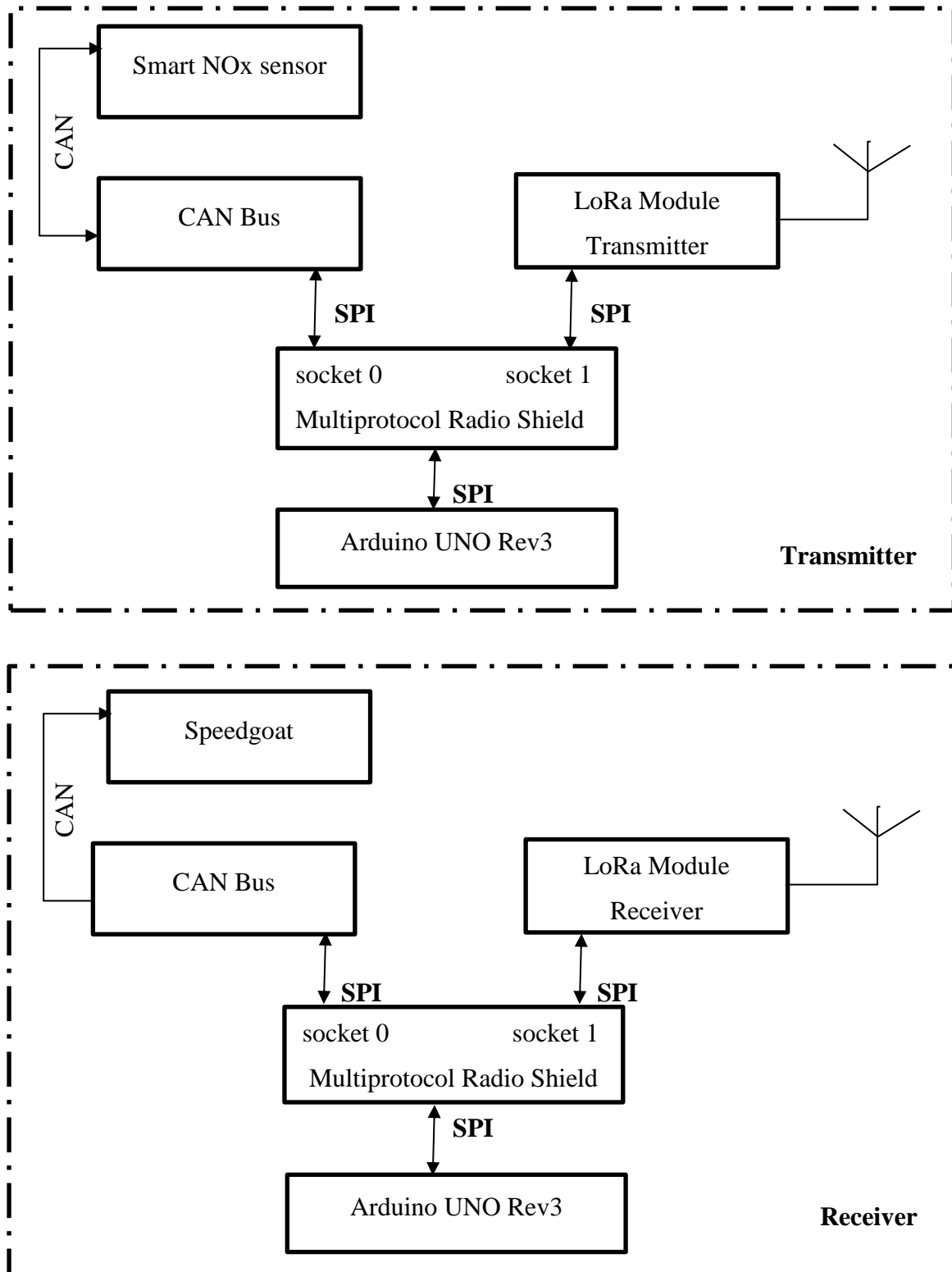


Figure 50. Block diagram for the hardware setup of LoRa-CAN bridge.

At the transmitter side, the Multiprotocol Radio Shield is connected over the Arduino board and the CAN Bus module is placed in socket 0 of the Multiprotocol Radio Shield while the LoRa module is placed in socket 1 of the Multiprotocol Radio Shield. The CAN Bus module is used to interface the Transmitter LoRa module with the smart NOx sensor using twisted pair cables (CAN High and CAN Low) to connect the smart NOx to the CAN Bus.

At the receiver side, the Multiprotocol Radio Shield is connected over the Arduino board and the CAN Bus module is placed in socket 0 of the Multiprotocol Radio Shield while the LoRa module is placed in socket 1 of the Multiprotocol Radio Shield. The Libelium CAN Bus module was used to interface the Receiver LoRa module with the speedgoat using twisted pair cables (CAN High and CAN Low) to connect the speedgoat to the CAN Bus.

3.2.9. The LoRa-CAN bridge Software



Figure 51. Smart NOx sensor and LoRa-CAN transmitter.

The programming was done on the Arduino IDE environment. At the transmitter side, the CAN Bus was programmed to send the 8 bytes hexadecimal heating signal “04h” to the smart NOx sensor to start heating the sensor to get the NOx CAN data frames from the smart NOx. The CAN Bus also receives the CAN Data sent from the smart NOx af-

ter it starts heating and transfers the data through SPI to the LoRa for wireless transmission. The smart NOx and LoRa-CAN transmitter setup is presented in figure 51.

At the receiver side, the LoRa is programmed to receive the smart NOx CAN Data and transfers the data through SPI to the CAN Bus. The CAN Bus is connected to the speedgoat using two twisted pair cable (CAN High and CAN Low) were the received CAN Data are analyzed. The Kvaser Leaf Light HS v2 USB can be used to view and debug the CAN data before connection to the speedgoat. The LoRa-CAN and Kvaser Leaf Light HS v2 USB setup is shown in figure 52.

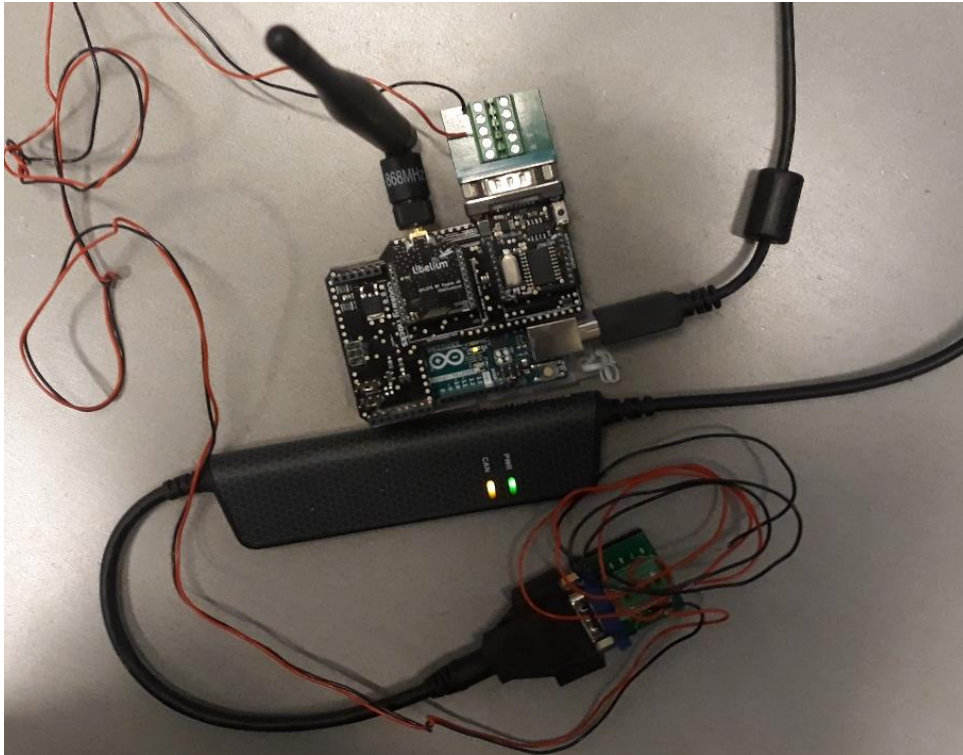


Figure 52. LoRa-CAN receiver and Kvaser Leaf Light HS v2 USB.

The flowcharts for the transmitter and receiver codes are presented in figure 53 and 54 respectively. The code of the transmitter contains the header file *CANLoRaTransmitterConfig.h* which is used to configure the LoRa modules to communicate with each other and to initialize the CAN Bus module. The header file *LoRaSendCANDataAES128.h* implements the required C functions to write the start heat-

ing command and extract the 8 data bytes from the received CAN frame through SPI interface. It sends the 8 bytes hexadecimal heating signal “04h” to start heating the smart NOx which is powered by 24V supply. The smart NOx sensor has a 29-bit CAN ID (0x18FEDF00 equivalent in decimal is 419356416) used to send the heat signal from the transmitter side through the CAN Bus to the smart NOx sensor. The header file *LoRaSendCANDataAES128.h* is used read the data from the smart NOx, compute a checksum for error detection, compute a *sender error detection number* (used as a preamble) which is the sum of the received smart NOx data plus checksum (this is compared with the *receiver error detection number* from the receiver module) and implement AES-128 encryption before transmission of the encrypted data using the LoRa module. Both the preamble and checksum are appended to the smart NOx data before encryption and transmission.

The code of the receiver contains the header file *CANLoRaReceiverConfig.h* used to configure the LoRa and CAN Bus modules. While the header file *LoRaRecvCANDataAES128.h* is used to receive the CAN frames from the transmitter module, implement AES-128 decryption on the received data, to compute a *receiver error detection number* (this is compared with the *sender error detection number* (preamble) from the transmitter module for error verification). It is the sum of the received data minus the preamble. The header file is also used to transmit the decrypted data to the speedgoat for analysis. The receiver code also has the header files *LoRaRecvPacketLossAES128.h* used for implementing the packet loss measurement and *LoRaRecvRSSIAES128.h* is combined with *measurement.h* for RSSI measurement.

3.2.10. Flowchart for codes and Viewing the CAN frames

The flowcharts for the BLE, XBee, WiFi and LoRa transmitter and receiver codes are presented in figure 53 and 54 respectively.

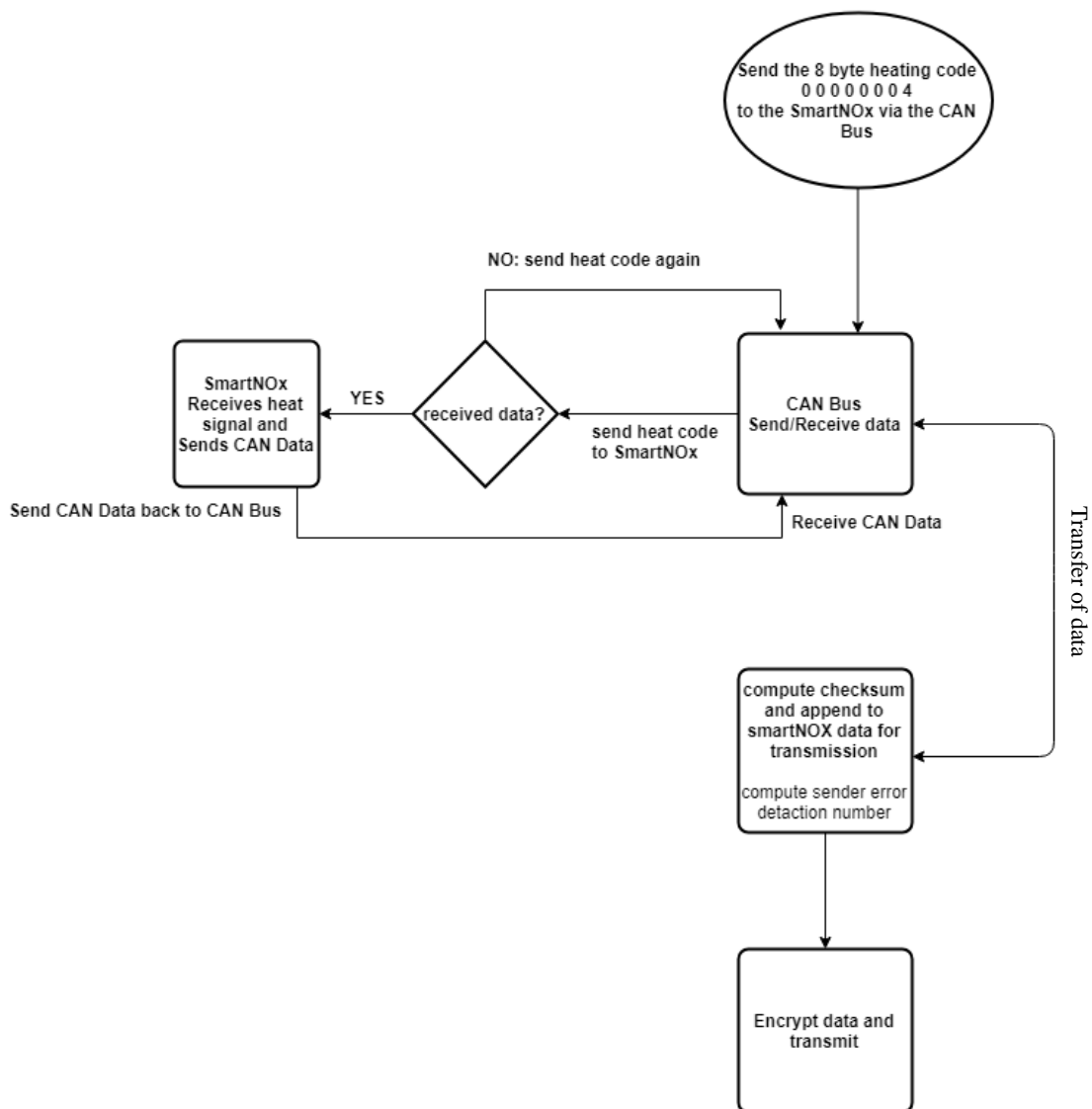


Figure 53. Flowchart for the BLE, XBee, WiFi and LoRa transmitter codes.

The flowchart in figure 53 and 54 represents the codes used to implement the BLE, XBee, WiFi and LoRa wireless solution. All wireless modules are programmed in the same pattern and step. The logic and process of the extraction of the smart NOx data, encrypting it, transmitting it, decryption of the data at the receiver side and sending the decrypted data to the speedgoat for analysis is fundamentally the same.

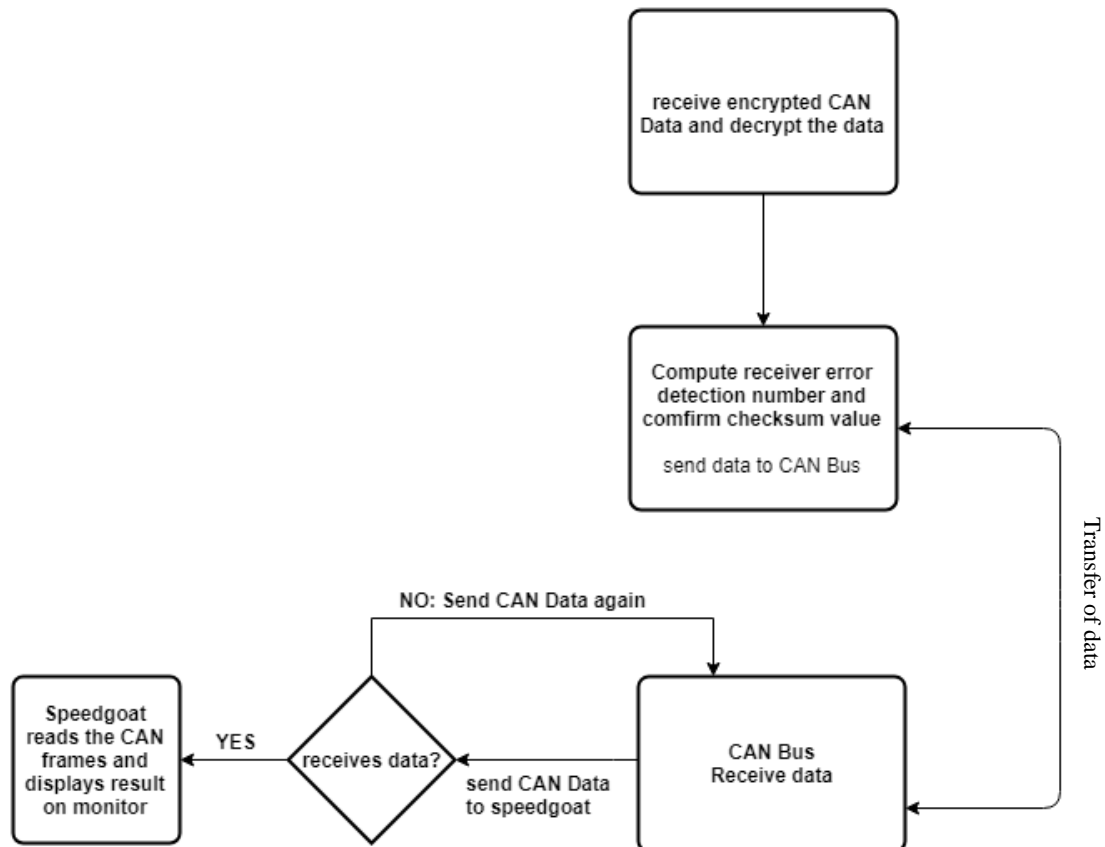


Figure 54. Flowchart for the BLE, XBee, WiFi and LoRa receiver codes.

The CAN data of the smart NO_x can be viewed and analyzed using Kvaser Leaf Light HS v2 USB connected to the CAN Bus of the receiver module for any of the wireless protocol and a BUSMASTER software.

The CAN frames are the 8 data byte that represents the measured values of the O₂ % and NO_x ppm. The data can be viewed in hexadecimal and decimal as illustrated in figure 55 and 56 respectively. The ID is the CAN ID of the smart NO_x used for communication with the CAN Bus. and the message ID shows the device ID of the CAN frames. The message ID and CAN ID are always the same for each device in this case the smart NO_x. The CAN frames are feed to the speedgoat to compute the O₂ % and NO_x ppm values for each receive message.

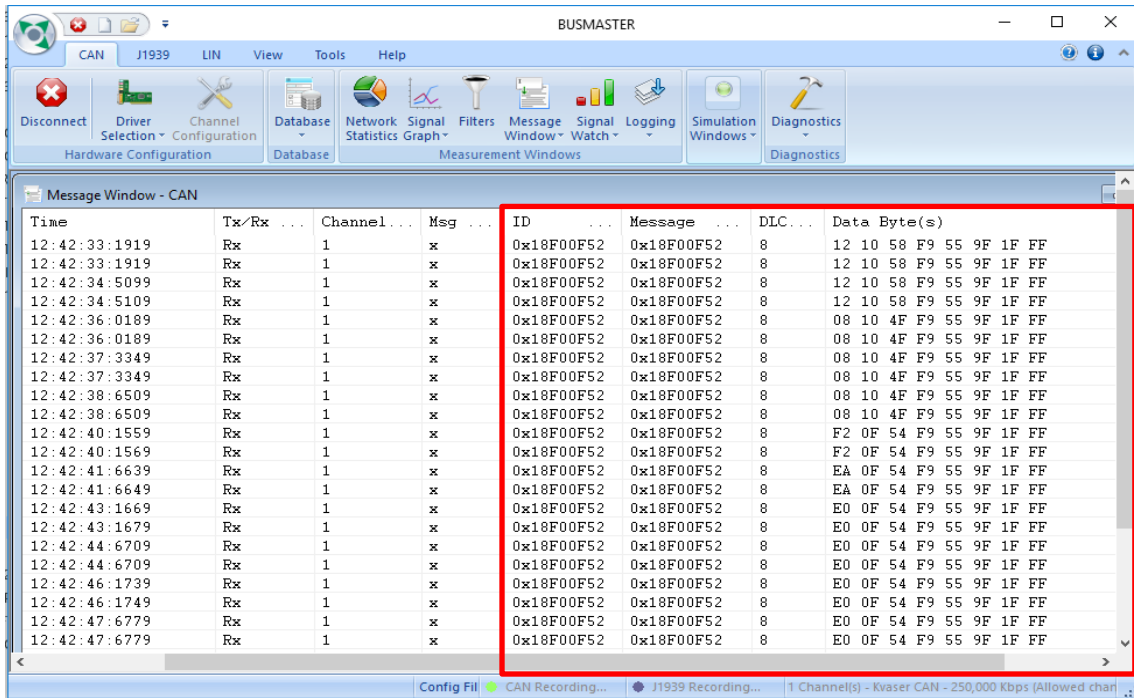


Figure 55. Hexadecimal View of the CAN frames.

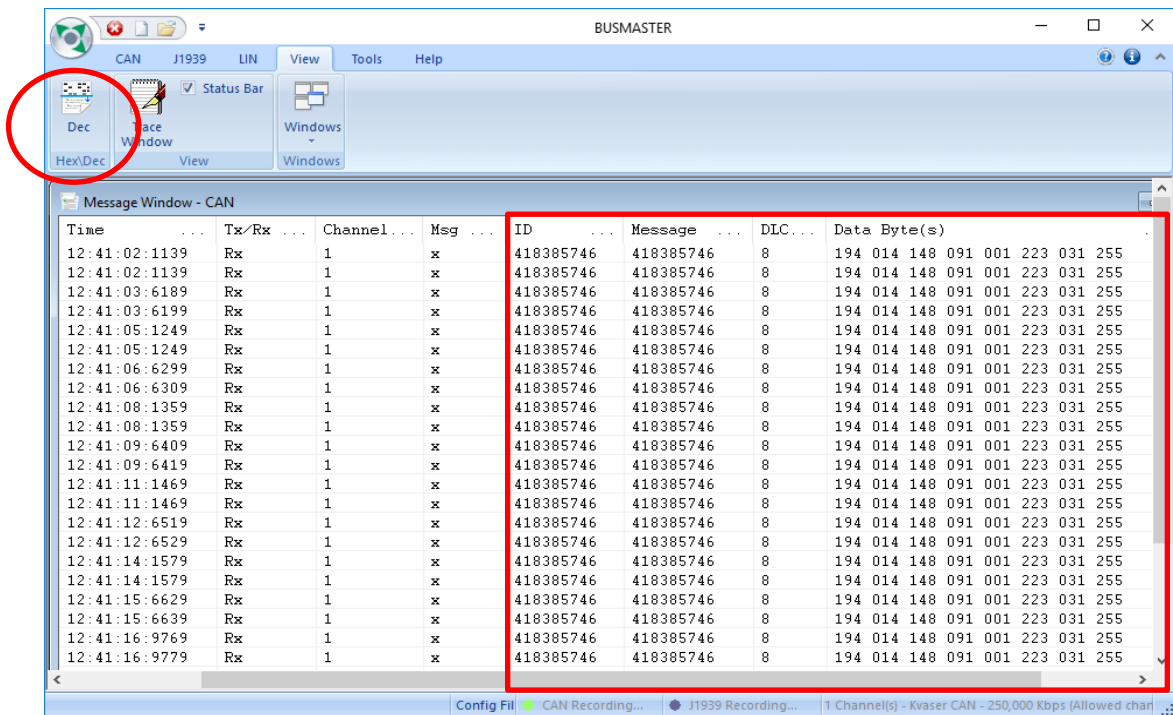


Figure 56. Decimal View of the CAN frames.

3.2.11. Connecting to the Speedgoat

The CAN Bus at the receiver side is connected to the speedgoat using twisted pair cables, CAN High and CAN Low, connected to the CAN I/O slot of the speedgoat. It uses CAN port 1 on the IO601 card of the speedgoat as shown in figure 35. This allows the speedgoat to read the CAN frames sent through the CAN Bus. The receive filter allows only extended CAN frames which has the same frame CAN ID sent by the client node CAN controller to be read and analyzed by the speedgoat.

The Simulink model represented in figure 36 continuously polls the client/receiver CAN module 4 times per second, extracts data bytes, calculates O₂ % and NO_x ppm and displays a continuously updated sliding graph on a monitor connected to the speedgoat. Figure 57 is an illustration of the sliding graph displayed on the monitor connected to the speedgoat.

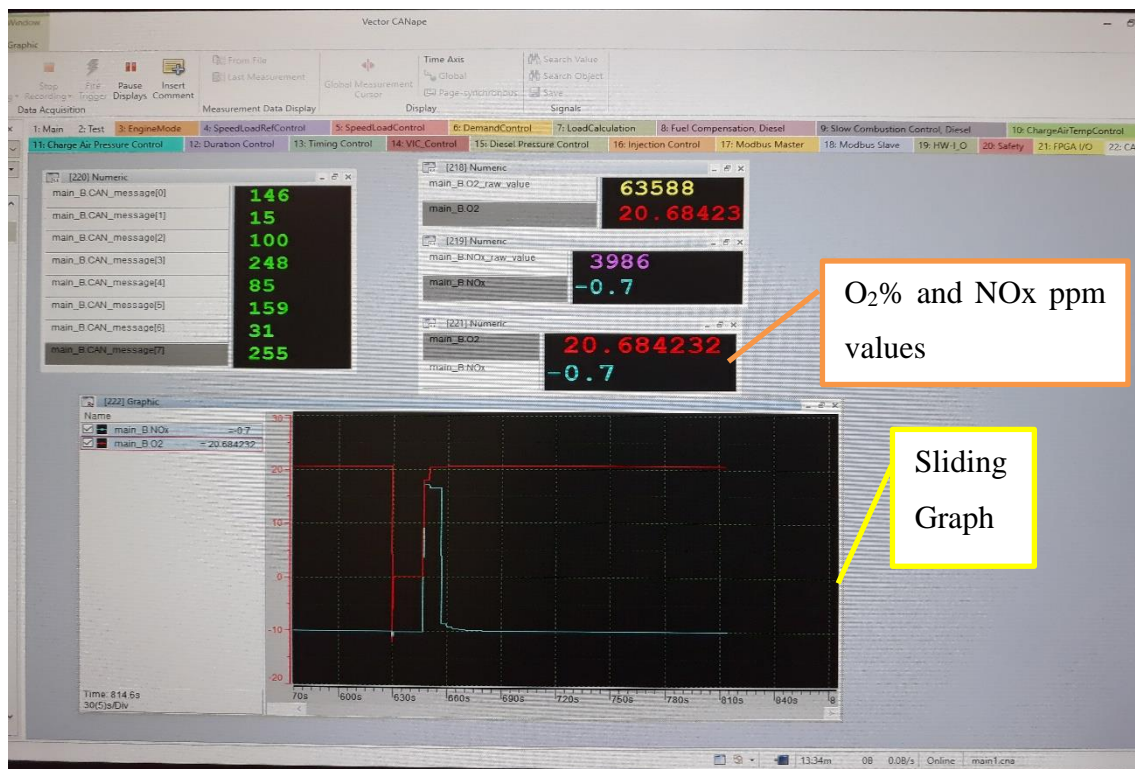


Figure 57. Continuously updated sliding graph for O₂% and NO_x ppm values.

3.3. Wireless Communication Performance Measure

Measuring the performance of each wireless protocol is important to evaluate the quality of the communication. The term Quality of Service (QoS) can be used to describe the expected services and performance of the wireless communication. The QoS is used to denote the communication properties such as communication reliability, security, probability of outage, communication performance (throughput, packet loss, latency and bit error rate), communication cost and priority. The QoS depends on the type of communication which can be categorized into two namely real time data stream from sensors and actuators and non-real time data stream from file UP/DOWN loads and internet exploring. (Elmusrati 2017.)

The communication performance analysis where done in three locations including Technobothnia and Tervahovi buildings in the University of Vaasa and VEBIC building where the speedgoat is installed. The purpose of this was to test the wireless communication in different environments. However, VEBIC represents the expected environment where the wireless solution is expected to be installed and utilized and therefore, comparisons will be mostly based on outcomes from VEBIC.

The measurements taken for each wireless protocol includes first the *packet loss* - 80 measurements were taken to measure the packet loss at distance of 5, 15, 25 and 30 meters for the Technobothnia and Tervahovi buildings in the University of Vaasa and at 12 meters in VEBIC building. The second is *RSSI* - 200 RSSI measurements were taken at intervals 5, 10, 15, 20, 25, and 30 meters for Technobothnia and Tervahovi buildings and at 12 meters in VEBIC. The third is *latency* – maximum and minimum latency values were taken and recorded. The fourth is *bit error rate* – making use of a checksum, the bit error rate can be analyzed. The checksum value is combined with the data at the transmitter side and sent along with the data. At the receiver side, the checksum is recomputed and compared with the checksum at the transmitter side to determine if the data is error free. The fifth is *power consumption analysis* – power consumption analysis for each wireless protocol was carried out and a comparison was made. The results of the performance measure are discussed and analyzed in chapter 4.

4. EXPERIMENT AND ANALYSIS

This chapter discusses the results of the experiment and simulations done as well as the comparison and analysis of the results for each wireless protocol implemented in this thesis. Communication performance such as packet loss, latency and bit error rate, RSSI and power consumption are discussed. In section 4.9, only the results from VEBIC's environment where the prototype will be installed was considered.

RSSI - The Received Signal Strength Indicator (RSSI) is the measure of the amount of power in a radio signal. It is measured in dBm. The quality of a communication link can be determined by measuring the signal strength at the receiving antenna. When a transmitter at a certain distant is moved towards a receiver, the received signal strength at the receiving antenna increases (lesser negative value). On the other hand, moving the transmitter away from the receiver makes the signal strength at the receiving antenna to decrease (greater negative value). RSSI with greater negative value indicates a weaker signal. This implies that, -30 dBm is better than -40 dBm. (DIGI 2017.) Receiver sensitivity is the lowest power level at which a receiver can detect an RF signal and demodulate the received data.

Bit error rate - In terms of digital transmission, a bit error can be defined as the number of bits received from a transmitted data stream through a communication medium that has been modified or altered as a result of interference, distortion, noise or bit synchronization errors. While bit error rate (BER) defines the number of bit errors per unit time. (Wikipedia 2018d.)

Latency - The delay in a network specifies the duration required to transmit a bit of data through the network from one node to another. It is usually measured in multiples or fractions of seconds. The delay otherwise called latency can be slightly different depending on the environment where the specific pair of communicating nodes are located. (Wikipedia 2018e.)

Packet loss – This is the measure of the amount of data packet that is lost before it reaches the receiver and it occurs when a data transmission error occurs, usually across wireless networks, or due to network congestion. Packet loss is a percentage measure of the packets lost with respect to packets sent. (Wikipedia 2018f.)

Power consumption – Because no equipment is 100% efficient, energy used by the equipment is more than the energy really needed. This happens as a result of energy lost as heat, vibrations and/or electromagnetic radiation. (Wikipedia 2018g.) Most wireless RF devices are usually battery-powered. This can introduce a challenge depending on the application and the location of the device may make it difficult and/or expensive to replace the battery. In many real-world scenarios, extending battery life is important and critical.

4.1. Details of Transmitted Payload and LCD Display for XBee-CAN Modules

In all four cases of the wireless protocols implemented, the transmitted payload is a 10 - byte hexadecimal data comprising of 1-byte preamble, 8-byte smart NO_x data and 1-byte checksum data as illustrated in figure 58.

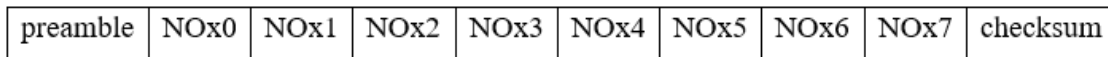


Figure 58. Transmitted Smart NO_x Payload.

The checksum is computed as shown in equation 15.

$$\text{NO}_{x0} + \text{NO}_{x1} + \text{NO}_{x2} + \dots + \text{NO}_{x7} = \text{checksum} \quad (15)$$

The preamble is computed as illustrated in equation 15.

$$\text{NO}_{x0} + \text{NO}_{x1} + \text{NO}_{x2} + \dots + \text{NO}_{x7} + \text{checksum} = \text{preamble} \quad (16)$$

Both the checksum and preamble are used to verify data integrity, that is, the data is error free (no bit error) and has not been altered in anyway.

On the XBee-CAN modules, an LCD display has been implemented to view the smart NOx sensor CAN ID and CAN data. (See APPENDIX 3.)

4.2. Bluetooth Low Energy (BLE)

In the BLE implementation, the communication performance analysis performed are discussed in the sub-sections 4.2.1 to 4.2.3. NA* = Not Applicable. The maximum distance available between transmitter and receiver was less than 15 meters.

4.2.1. BLE RSSI Values

The maximum and minimum measured RSSI values for Technobothnia and VEBIC are presented in table 12 and table 13 respectively. At each position 200 RSSI measurements have been taken. The BLE module has a receiver sensitivity of -103 dBm and maximum range of 100 meters.

Table 12. BLE maximum and minimum RSSI measurement in Technobothnia.

Distance in meters	5	10	15	20	25	30
Minimum RSSI value (dBm)	-49	-62	-48	-55	-71	-63
Maximum RSSI value (dBm)	-39	-44	-43	45	-49	-52

Table 13. BLE maximum and minimum RSSI measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Minimum RSSI value (dBm)	-69	-81	-83	NA*	NA*	NA*
Maximum RSSI value (dBm)	-47	-52	-56	NA*	NA*	NA*

Table 12 and table 13 shows that the RSSI values decreases (greater negative value) as the distance increases. This corresponds to the RSSI theory but in an ideal case the RSSI values would decrease linearly. This is however more obvious in table 13 for the VEBIC building. The nature of the application of this project does not require very large distance (maximum required distance is 30 meters), therefore the RSSI at the specified maximum distance of the module was not tested.

4.2.2. BLE Packet Loss

A total of 80 measurements (80 packets sent) were taken for the packet loss in Technobothnia buildings in the University of Vaasa and in VEBIC building at specified distances as presented in table 14 and table 15 respectively.

Table 14. BLE maximum and minimum Packet Loss measurement in Technobothnia.

Distance in meters	5	10	15	20	25	30
Total Packet Loss	0%	0%	0%	0%	0%	0%
Total Packet Received	100%	100%	100%	100%	100%	100%

Table 15. BLE maximum and minimum Packet Loss measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Total Packet Loss	0%	0%	0%	NA*	NA*	NA*
Total Packet Received	100%	100%	100%	NA*	NA*	NA*

In the experiment, table 14 and 15 presents the results. No packet loss was recorded during the measurements in both environments (Technobothnia and VEBIC). It had a 0% packet loss. This implies all sent packets were received.

4.2.3. BLE Latency

The maximum and minimum latency measurement noticed for the BLE implementation irrespective of the location are illustrated in the table 16.

Table 16. BLE maximum and minimum latency measurement in milliseconds.

Minimum latency value (ms)	24
Maximum latency value (ms)	60

4.3. XBee (IEEE 802.15.4)

In this XBee implementation, the communication performance analysis performed are discussed in the sub-sections 4.3.1 to 4.3.3. NA* = Not Applicable. The maximum distance available between transmitter and receiver was less than 15 meters.

4.3.1. XBee RSSI Values

The maximum and minimum measured RSSI values for Technobothnia and VEBIC are presented in table 17 and table 18 respectively. At each position 200 RSSI measurements have been taken. The XBee module has a receiver sensitivity of -100 dBm and maximum range of 750 meters.

In table 17 and table 18, the RSSI values decreases (greater negative value) as the distance increases. This corresponds to the RSSI theory but in an ideal case the RSSI values would decrease linearly. The nature of the application of this project does not require very large distance (maximum required distance is 30 meters), therefore the RSSI at the specified maximum distance of the module was not tested.

Table 17. XBee maximum and minimum RSSI measurement in Technobothnia.

Distance in meters	5	10	15	20	25	30
Minimum RSSI value (dBm)	-45	-47	-52	-56	-60	-66
Maximum RSSI value (dBm)	-40	-43	-46	-51	-53	-57

Table 18. XBee maximum and minimum RSSI measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Minimum RSSI value (dBm)	-53	-54	-60	NA*	NA*	NA*
Maximum RSSI value (dBm)	-42	-47	-52	NA*	NA*	NA*

4.3.2. XBee Packet Loss

A total of 80 measurements (80 packets sent) were also taken for the packet loss in Technobothnia buildings in the University of Vaasa and in VEBIC building at specified distances as presented in table 19 and table 20 respectively.

Table 19. XBee maximum and minimum Packet Loss measurement in Technobothnia.

Distance in meters	5	10	15	20	25	30
Total Packet Loss	0%	0%	0%	0%	0%	0%
Total Packet Received	100%	100%	100%	100%	100%	100%

Table 20. XBee maximum and minimum Packet Loss measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Total Packet Loss	0%	0%	0%	NA*	NA*	NA*
Total Packet Received	100%	100%	100%	NA*	NA*	NA*

In the experiment, table 19 and 20 records no packet loss during the measurements in both environments (Technobothnia and VEBIC). It had a 0% packet loss. This implies all sent packets were received.

4.3.3. XBee Latency

The maximum and minimum latency measurement noticed for the XBee implementation irrespective of the location are illustrated in the table 21.

Table 21. XBee maximum and minimum latency measurement in milliseconds.

Minimum latency value (ms)	102
Maximum latency value (ms)	106

4.4. WIFI (IEEE 802.11 b/g/a)

In this WIFI implementation, the communication performance analysis performed are discussed in the sub-sections 4.4.1 to 4.4.3. NA* = Not Applicable. The maximum distance available between transmitter and receiver was less than 15 meters.

4.4.1. WIFI RSSI Values

The maximum and minimum measured RSSI values for Technobothnia and VEBIC are presented in table 22 and table 23 respectively. At each position 200 RSSI measurements have been taken. The WIFI module has a receiver sensitivity of -94 dBm to -70dBm depending on if you use b/g/n and maximum range of <300 meters.

Table 25. WIFI maximum and minimum Packet Loss measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Total Packet Loss	0%	0%	0%	NA*	NA*	NA*
Total Packet Received	100%	100%	100%	NA*	NA*	NA*

In the experiment, table 24 and 25 presents the results. No packet loss was recorded during the measurements in both environments (Technobothnia and VEBIC). It had a 0% packet loss. This implies all packets sent were received.

4.4.3. WIFI Latency

The maximum and minimum latency measurement noticed for the WIFI implementation irrespective of the location are illustrated in the table 26.

Table 26. WIFI maximum and minimum latency measurement in milliseconds.

Minimum latency value (ms)	1169
Maximum latency value (ms)	1213

4.5. LoRa (Long Range)

In this LoRa implementation, the communication performance analysis performed are discussed in the sub-sections 4.5.1 to 4.5.3. NA* = Not Applicable. The maximum distance available between transmitter and receiver was less than 15 meters.

4.5.1. LoRa RSSI Values

The maximum and minimum measured RSSI values for Technobothnia and VEBIC are presented in table 27 and table 28 respectively. At each position 200 RSSI measure-

ments have been taken. The LoRa module has a receiver sensitivity of -134 dBm and maximum range of 22 kilometers.

Table 27. LoRa maximum and minimum RSSI measurement in Technobothnia.

Distance in meters	5	10	15	20	25	30
Minimum RSSI value (dBm)	-45	-75	-62	-57	-76	-82
Maximum RSSI value (dBm)	-36	-45	-44	-44	-51	-55

Table 28. LoRa maximum and minimum RSSI measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Minimum RSSI value (dBm)	-64	-58	-58	NA*	NA*	NA*
Maximum RSSI value (dBm)	-46	-52	-50	NA*	NA*	NA*

In table 27 and table 28 the RSSI values seems to decrease (greater negative value) as the distance increases. This corresponds to the RSSI theory but in an ideal case the RSSI values would decrease linearly. The nature of the application of this project does not require very large distance (maximum required distance is 30 meters), therefore the RSSI at the specified maximum distance of the module was not tested.

4.5.2. LoRa Packet Loss

A total of 80 measurements (80 packets sent) were also taken for the packet loss in Technobothnia buildings in the University of Vaasa and in VEBIC building at specified distances as presented in table 29 and table 30 respectively.

Table 29. LoRa maximum and minimum Packet Loss measurement in Technobothnia.

Distance in meters	5	15	25	30
Total Packet Loss	5%	5%	7%	5%
Total Packet Received	95%	95%	93%	95%

Table 30. LoRa maximum and minimum Packet Loss measurement in VEBIC.

Distance in meters	5	10	13	20	25	30
Total Packet Loss	0%	0%	0%	NA*	NA*	NA*
Total Packet Received	100%	100%	100%	NA*	NA*	NA*

In the experiment, table 29 records some packet loss during the measurements in Technobothnia with maximum packet loss percentage of 7% and no packet loss in VEBIC (see table 30).

4.5.3. LoRa Latency

The maximum and minimum latency measurement noticed for the LoRa implementation irrespective of the location are illustrated in the table 31.

Table 31. LoRa maximum and minimum latency measurement in milliseconds.

Minimum latency value (ms)	2102
Maximum latency value (ms)	2106

4.6. Bit Error Check for all wireless protocols

Bit error check for all the wireless protocols were implemented in the same way. The payload is a 10-byte data payload. The payload at the receiver is checked for error using a 1-byte preamble (ErrorDectNum) and 1-byte checksum computed and appended to the 8-byte smart NOx data before transmission. The checksum is the sum of the 8-byte smart NOx data only while the preamble is computed from the sum of the 8-byte smart NOx data plus the checksum value, that is, $\text{smartNOx}[8] + \text{checksum}[1] = \text{preamble}[1]$. If the preamble and checksum of the sender is the same as the preamble and checksum of the receiver, the data is error free. Alternatively, if the sum of the $\text{smartNOx}[8] + \text{checksum}[1]$ at the receiver side equals the preamble at the receiver side of the same

payload been considered, the data is error free. The sample output of the transmitter and receiver code is illustrated in APPENDIX 4.

4.7. Security Implementation

The security implementation used in all four cases of the wireless protocols is the AES-128 encryption/decryption. The encryption of the data is done using AES encryption mode CBC. The security implementation was done at the code level. That is, a code was written to implement the AES128 encryption on the data. However, only the XBee module also has the capability of implementing AES128 encryption on the module itself. In this case, only the receiver XBee module with the correct decryption key can receive and understand the transmitted encrypted data. With this, the XBee module has two levels of encryption, one at the XBee module level and the second at the coding level.

4.8. Power Consumption

The battery life (power consumption) analysis of the wireless modules are computed by dividing the battery capacity (in mAh) by the total average current (in μA). This formula is illustrated in equation 14. (Digi-Key 2018.)

$$\text{Battery Life}[H] = \frac{\text{Battery Capacity}[mAh]}{\text{Total Average Current } [\mu A]} \quad (14)$$

The main element here is the total average current; it is the sum of all events (steady-state and periodic) as well as the battery self-discharge. The formula is presented in equation 15. (Digi-Key 2018.)

$$\begin{aligned} I(\text{total average current})[\mu A] = \\ I(\text{steady - state operation average})[\mu A] + \\ I(\text{periodic events average})[\mu A] \end{aligned} \quad (15)$$

where steady-state current is the sleep current and the periodic event current are the RX /TX currents.

Applying the formula in equation 14 and 15, we can calculate the battery life for the various wireless protocol modules applied in this thesis. In all four cases (BLE, XBee, WIFI and LoRa, a 3.7 V battery with 6600mAh battery capacity is used. The battery life for each module is computed as illustrated below. The battery life will be computed for the transmitter and receiver modules separately as these modules have only one event (transmitting or receiving). From equation 15 we first compute the total average current for the transmitter module and then from equation 14 the battery life for the transmitter and receiver modules are computed. Table 32 presents the computed values for the expected battery life for the wireless modules based on the transmit current, receive current specifications in their respective datasheets.

Table 32. Computed Battery Life of Transmitter and Receiver Modules

Parameters	BLE	XBee	WIFI	LoRa
Sleep current (4 μA)	0.4 μ A	<10 μ A	<100 μ A	NA*
Transmit current (mA)	36 mA	215 mA	350 mA	NA*
Receive current (mA)	8 mA	55 mA	130 mA	NA*
I(total average current)[μA]	3600.4	215010	350100	NA*
Receiver Battery Life[Hours]	825	119	51	NA*
Transmitter Battery Life[Hours]	183	31	19	NA*

NA = Not Applicable > LoRa Module - The power consumption specifications of the LoRa module is not mentioned in the datasheet. However, there is a recommendation to use solar or mains electricity to power the module.

4.9. Comparing the Wireless Solutions Based on the Analysis of Results

Table 33 is a comparison of the wireless protocols based on the results gotten from sections 4.1 to 4.4. These comparisons are some key considerations that should influence the choice of wireless protocols for a specific application.

Table 33. Comparison of the wireless protocols based on the analysis of results.

Considerations	BLE		XBee (802.15.4)		WIFI		LoRa	
	Data Sheet	Expt*	Data Sheet	Expt*	Data Sheet	Expt*	Data Sheet	Expt*
RSSI VEBIC	-103 dBm	-83 dBm	-100 dBm	-60 dBm	-94 dBm	-70 dBm	-134 dBm	-64 dBm
RSSI Technobothnia	-103 dBm	-71 dBm	-100 dBm	-66 dBm	-94 dBm	-100 dBm	-134 dBm	-82 dBm
Packet Loss %	N/M*	0%	N/M*	0%	N/M*	0%	N/M*	0%
Min. Latency Max. Latency	N/M	24ms 60ms	N/M	102ms 106ms	N/M	1169ms 1213ms	N/M	2102m s 2106m s
Power Consumption/ Battery Capacity	Tx 36m A Rx 8mA	3.7V 6600 mAh battery	Tx 215mA Rx 55mA	3.7V 6600 mAh battery	Tx 350mA Rx 130mA	3.7V 6600 mAh battery	NM	3.7V 6600 mAh battery
Battery Life [Hours]		Tx 183 Rx 825		Tx 31 Rx 119		Tx 19 Rx 51	NM	NM
Security	AES 128	AES 128 at code level	AES 128	AES 128 at Mod- ule and code level	AES 128/256 , SSL3/T LS1, HTTPS, RSA, 3DES, WEP, WPA and WPA2	AES 128 at code level	AES 128	AES 128 at code level

*LOS = Line of Sight. Maximum data rates are often not available at the longest range

* Expt = Experiment Results; *N/M = Not Mentioned; *N/A = Not Measured

The theoretical knowledge is that a greater negative value (in dBm) indicates a weaker signal.

Technobothnia RSSI - From the table 33, we can deduce that the wireless protocol with the best theoretical minimum RSSI value is the WIFI with -94dBm and next is the XBee with -100dBm based on their datasheet. However, from the experimental results, XBee shows the best minimum RSSI value of -66 dBm. Figure 59 is an illustration of the measured minimum and maximum RSSI value for all the wireless protocol done in Technobothnia.

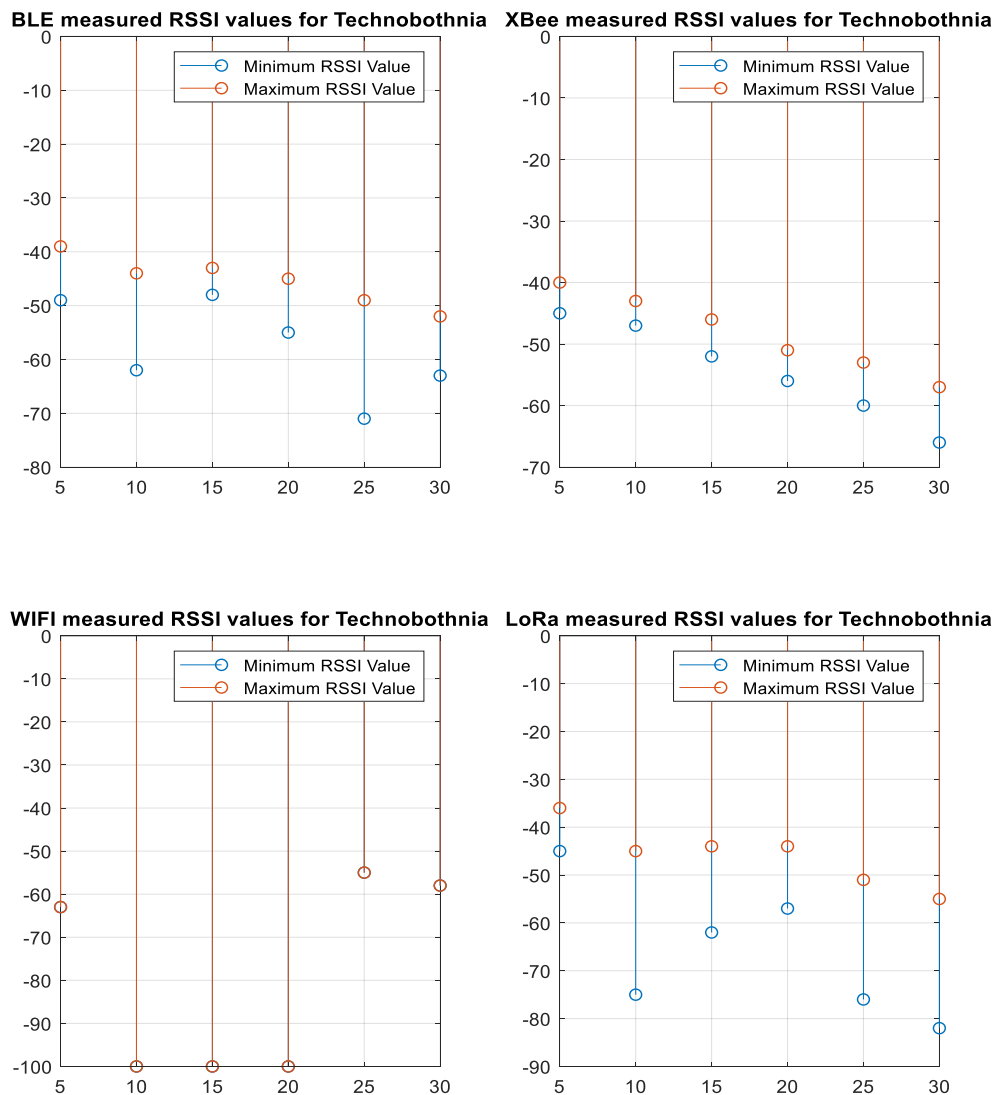


Figure 59. RSSI Measurements for all the wireless protocol in Technobothnia.

VEBIC RSSI - From the table 33, we can deduce that the wireless protocol with the best theoretical minimum RSSI value is the WIFI with -94dBm and next is the XBee with -100dBm based on their datasheet. However, from the experimental results, XBee shows the best minimum RSSI value of -60dBm. Figure 60 is an illustration of the measured minimum and maximum RSSI value for all the wireless protocol done in VEBIC.

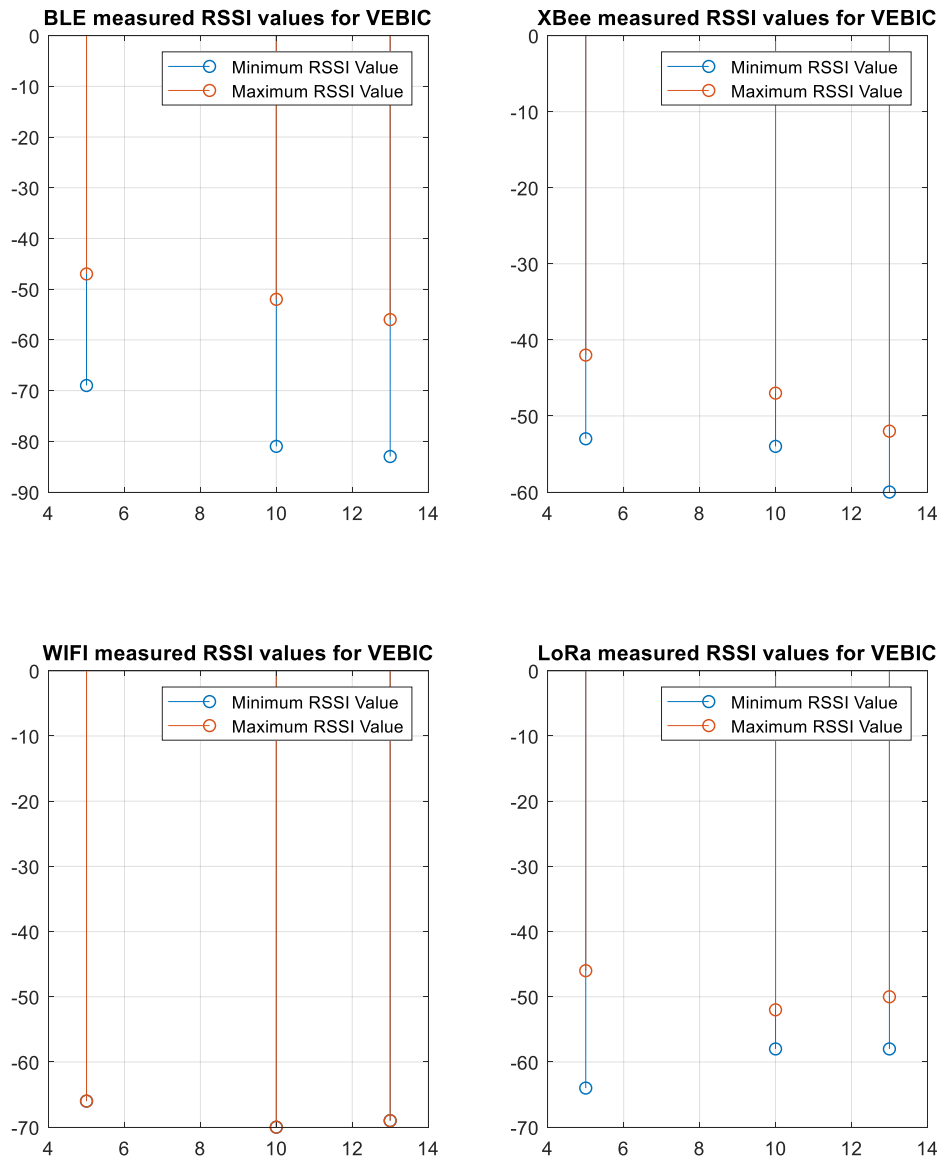


Figure 60. RSSI Measurements for all the wireless protocol in VEBIC.

Packet Loss – There were no packet loss for any of the four modules at the maximum test range of 30 meters and 15 meters.

Security – Security implementation can be made on all wireless protocols. The XBee module has the extra feature of enabling AES128 encryption on the XBee module itself. In other modules (BLE, WIFI and LoRa), data encryption is an optional feature provided by an API library. However, WIFI module also provide the feature of installing Secure Sockets Layer (SSL), it requires installing the corresponding certificate, created by a CA (Certification Authority). This makes it more complex than the feature of the XBee module. Likewise, the BLE module uses AES-128 link layer encryption for encrypting the connection to make the connection processes secure. The data however is not encrypted. Since the XBee module provides the feature of AES 128 encryption on the module and an implementation of AES 128 encryption API library, this makes the XBee module to provide a two level of security of the data compared to the other wireless modules. The AES 128 encryption on the XBee modules is done during the configuration of the XBee modules, while the AES128 encryption API libraries provided by Arduino was used along with C program functions to provide encryption of the data at the coding level. LoRa was also implemented on the Arduino development board. In the case of WIFI and BLE, Arduino AES 128 API library was modified to be compatible with the Waspnote development board on which they were implemented.

Power Consumption - In table 36, the BLE shows a better battery life, next is the XBee and the third is the WIFI module. The LoRa module specification recommends powering the LoRa with solar or mains electricity. This is not practicable when dealing with wireless sensors depending on the location of the sensor and where the LoRa module is to be placed. It also implies that LoRa is not suitable for projects requiring high data-rate and very frequent transmissions for example, every 10 seconds. (Libelium 2017b.)

However, the advantage of the LoRa module over the other wireless solution is that LoRa is the best option when dealing with very wide networks, with long-range links. Other communication modules cannot get more than few km. The 3 features of LoRa such as good sensitivity, low path loss, good obstacle penetration gives LoRa the ability

of transmitting and receiving in very long-range. It significantly reduces the size of the backbone network such as repeaters, gateways or concentrators.

Maximizing battery life is critical in wireless sensor applications. There are several ways to maximize battery life such as reducing the data transmission rates and putting the modules into a cycle where they sleep for one second and then wake for one second to transmit the data before sleeping again. This could double the battery life to two days specifically in the case of the BLE and XBee modules. Enhancing the cyclically sleep and wake event of the BLE and XBee modules could potentially prolong the battery life for years.

From the experiments, the XBee had a better RSSI value and security feature over the other wireless modules used. These features and the good performance in the packet loss test, the ability to enhance the battery life and a better penetrating capability and range when compared to the BLE leads to the recommendation of implementing this wireless protocol as the main choice in the Wärtsilä smart NO_x sensor case study. Table gives an overview summary of the results.

4.10. Viewing O₂% and NO_x ppm Values on the Speedgoat

The CAN frame O₂% and NO_x ppm values are sent to the speedgoat through the CAN Bus (CAN High and CAN Low). The general view of the expected results on the monitor connected to the speedgoat is illustrated in figure 61.

From figure 61, the NO_x-concentration detected by the NO_x-Sensor is transmitted. The transmission is in 0.05 ppm NO_x/bit + 200 ppm. That means for the value 3966 the computed ppm is $3966 * 0.05 - 200 = -1.7$ ppm.

Likewise, in figure 61 for signal of the actual oxidation factor (O₂%), the transmission is in 0.000514%/bit + 12%. That is, the O₂% can be computed from the read value of 63588 to correspond to $63792 * 0.000514 - 12 = 20.78908\%$ or 21%.

These O₂% and NO_x ppm values corresponds to the expected values for any measurement as illustrated in the smart NO_x sensor datasheet provided by Wärtsilä.

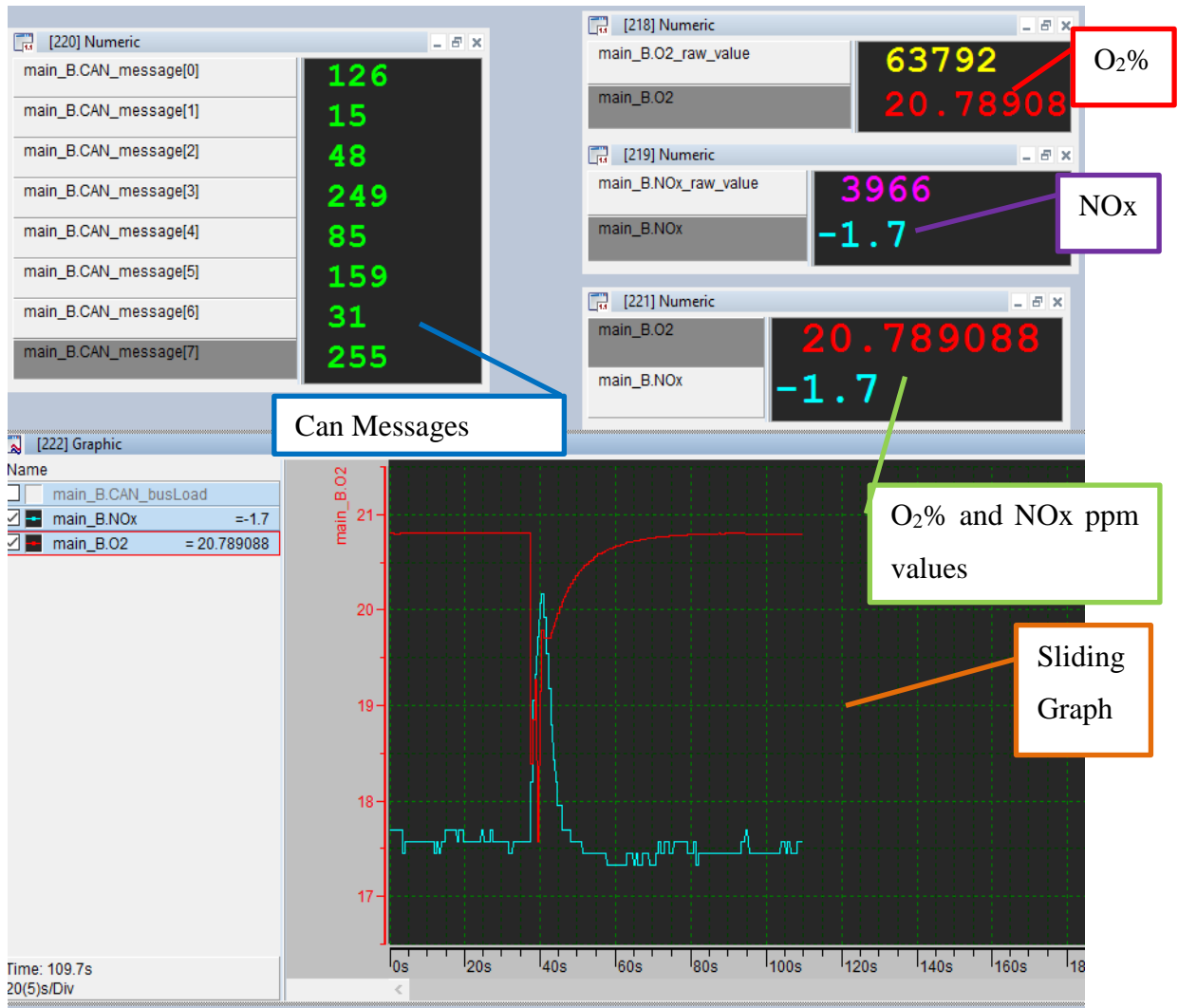


Figure 61. Continuously updated sliding graph for O₂% and NO_x ppm values.

4.11. SmartNO_x + XBee-CAN Module Test on Wärtsilä W4L20 Diesel Engine

A smart NO_x sensor was installed on the Wärtsilä W4L20 Diesel Engine and the O₂% and NO_x ppm values were measured and compared with the readings from the SICK MCS100E. The MCS100E HW is an analyzer system used for extractive measurement of up to eight (8) active gas components from an engine. It also can be used for measur-

ing water-soluble components such as HCl and NH₃. (SICK 2018.) Table 34 illustrates the comparison of the values from SICK and smart NO_x sensor/speedgoat for the Wärtsilä W4L20 Diesel Engine for different operation modes (engine is idle, running without load and running with a load).

Table 34. Comparison of the values from SICK and Smart NO_x sensor for the Wärtsilä W4L20 Diesel Engine for different operation modes.

Measurement Device	Engine is Idle		Engine running without load		Engine running with load	
	NO _x ppm	O ₂ %	NO _x ppm	O ₂ %	NO _x ppm	O ₂ %
SICK MCS 100E	NA	NA	162	15.09	649	12.65
Smart NO_x and Speedgoat	-2.1	20.89	167	16.18	624.1	12.39

From table 34, it can be deduced that the smart NO_x and Speedgoat are giving the readings close to the values from SICK|MCS 100E.

Figure 62 illustrates the results on the speedgoat when the Wärtsilä W4L20 Diesel Engine is idle (not running). Figure 63 is the engine status for when is it running without load and figure 64 gives the sliding graph of the O₂% and NO_x ppm values as illustrated in figure 65. Similarly, figure 66 is the engine status for when is it running with load and figure 67 gives the sliding graph of the O₂% and NO_x ppm values as illustrated in figure 68.

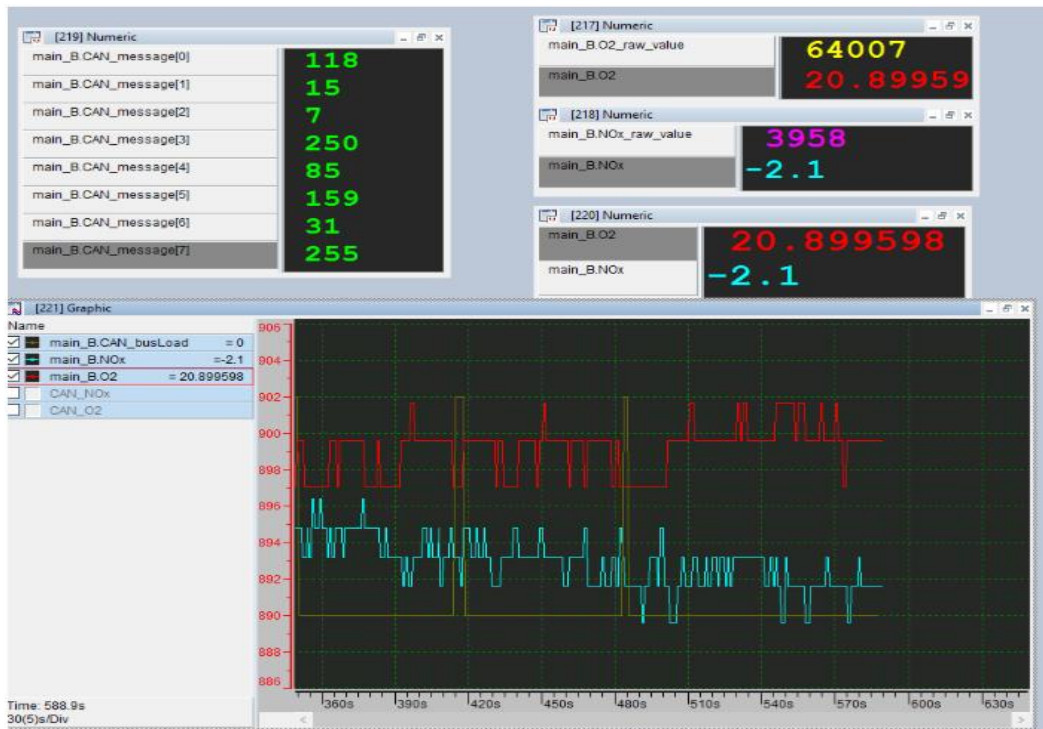


Figure 62. Speedgoat result when Wärtsilä W4L20 Diesel Engine is idle.

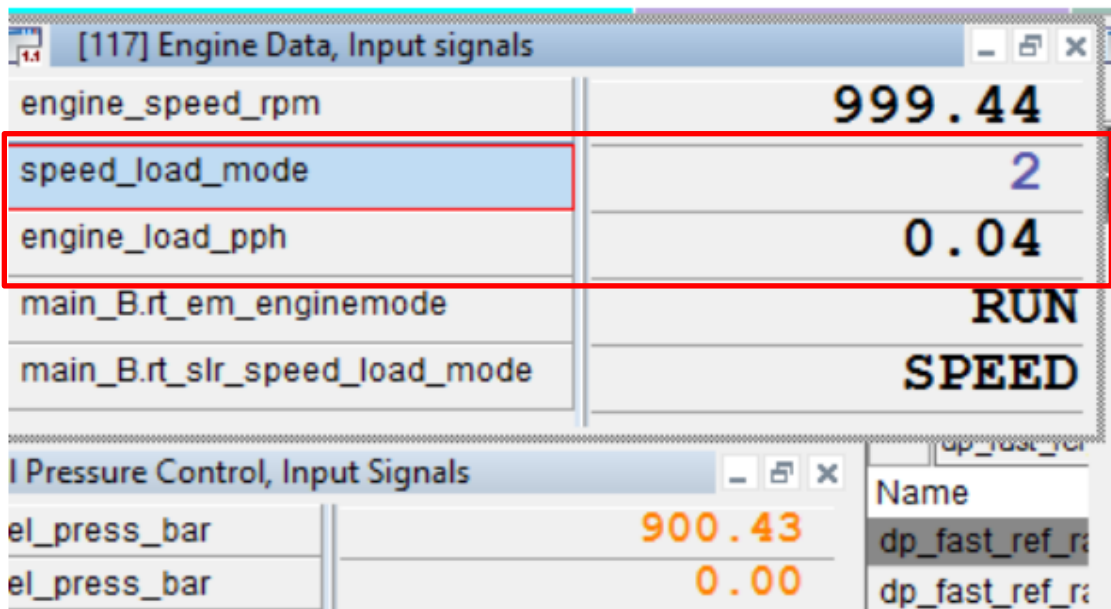


Figure 63. Wärtsilä W4L20 Diesel Engine is running without load

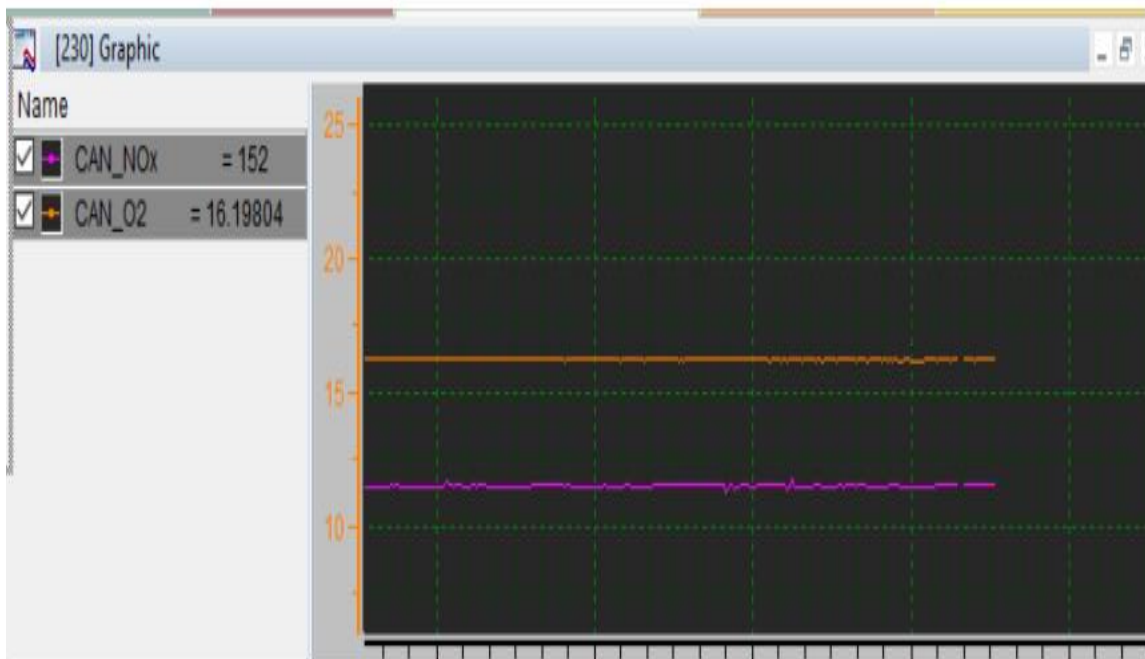


Figure 64. Speedgoat sliding graph of the O₂% and NO_x ppm values.

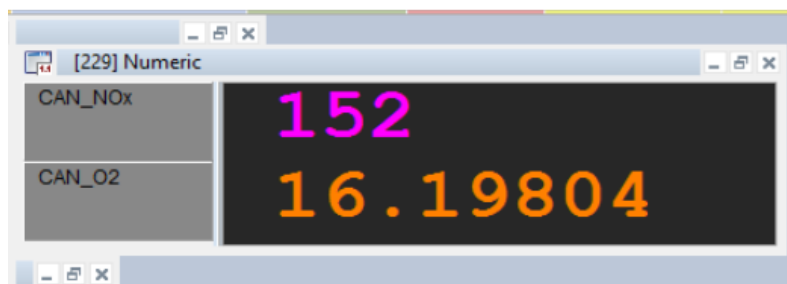


Figure 65. Speedgoat results for O₂% and NO_x ppm values.

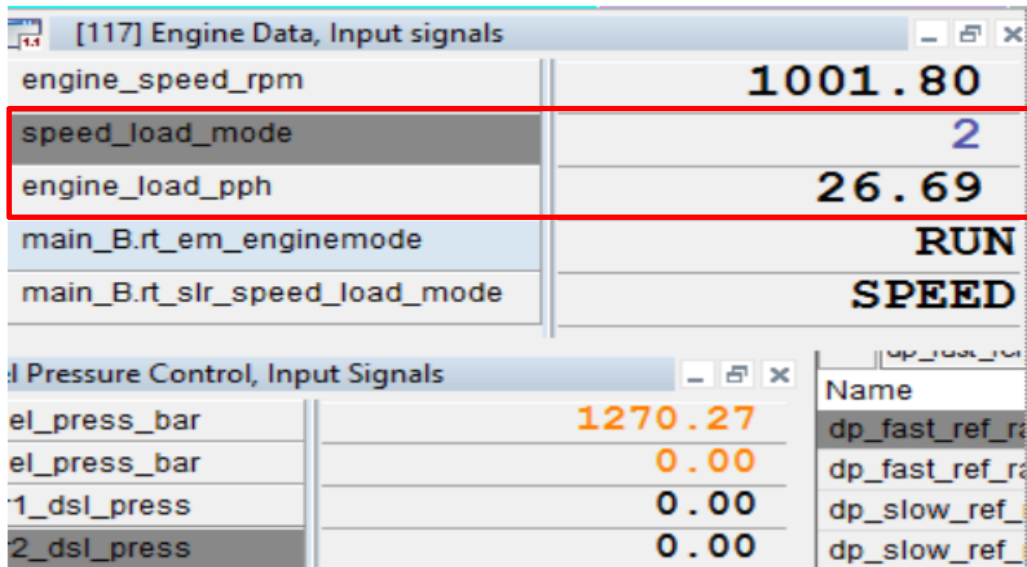


Figure 66. Wärtsilä W4L20 Diesel Engine is running with load

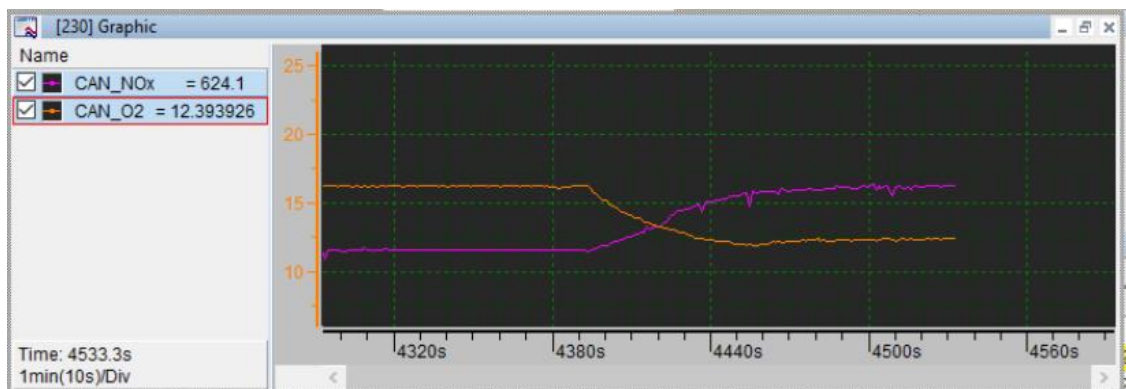


Figure 67. Speedgoat sliding graph of the O₂% and NO_x ppm values.

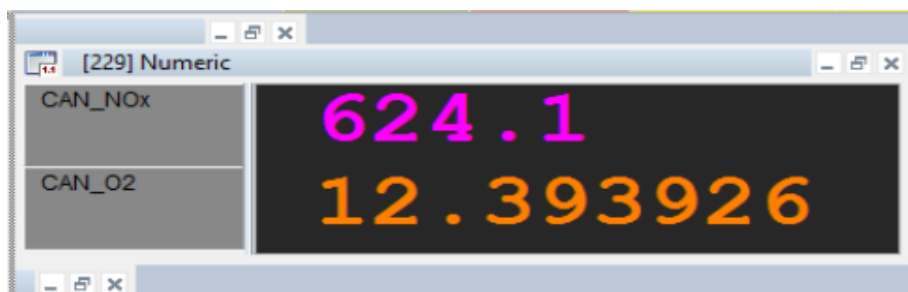


Figure 68. Speedgoat results for O₂% and NO_x ppm values.

4.12. Applying Additive Manufacturing (3D printing) to the Designed Prototype

Additive manufacturing is a developing technology in industrial production which facilitates the creation parts and systems that are lighter and stronger. There are several different types of additive manufacturing such as 3D printing, rapid prototyping and direct digital manufacturing (DDM). The term “additive manufacturing” is used to relate to the technologies that creates three-dimensional objects one superfine layer at a time with each successive layer bonding to the preceding layer of melted or partially melted material. The different materials or substances used includes metal powder, thermoplastics, ceramics, composites, glass and even edibles substances such as chocolate. (GE Additive 2018.)

Computer-aided-design (CAD) software are used to digitally define the objects by creating a file with .stl extension which essentially "slices" the object into ultra-thin layers. The various steps from the .stl file to the printed 3D object has been revolutionizing manufacturing. This has eliminated the intermediary steps such as creation of molds or dies, that cost time and money. These advancements in this technology have made a far more widespread use of additive manufacturing and still offers exciting possibilities for future development. (GE Additive 2018.)

Additive Manufacturing is applied in this project to design the protective casing for the designed prototype developed for the wireless communication between the smart NOx and the speedgoat. Two protective casings were designed, and 3D printed. The transmitter module and the receiver module were encapsulated in each of these protective casings respectively as illustrated in APENDIX 4. The Material used in printing the protective casings is PLA filament. The 3D printer used for the printing is the PRUSA and MINIFactory. Both used the following settings during printing: printing temperature of 200°C for the extruder and 60°C for the bed plate and an infill of 20%. The 3D printed protective casings are presented in APPENDIX 5 and 6. While the installed XBee-CAN Receiver/Transmitter in the 3D printed protective casings are illustrated in figure 69.

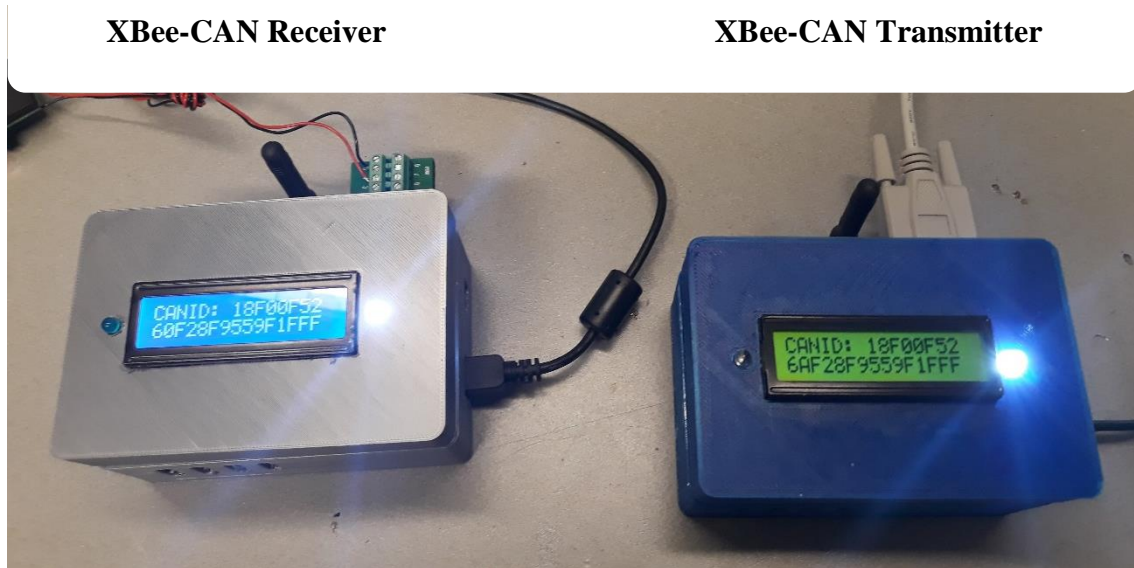


Figure 69. XBee-CAN Receiver/Transmitter in 3D printed protective casings.

5. CONCLUSION AND FUTURE WORK

This thesis work was carried out in line with the expectation of Wärtsilä to investigate the possibility of replacing the existing wired CAN bus connection between the smart NOx sensor and the rapid control prototyping system speedgoat and possibly in the future the engine control unit (ECU) with a wireless communication solution. Implementations and analyses of some wireless protocols has been done with the attempts to come up with a comparison and possible recommendation based on some criteria like transmission in industrial environments, packet loss, RSSI, bit error rate, reliability and security of the wireless solution etc. The wireless protocols implemented includes BLE, XBee (ZigBee), WIFI and LoRa (Long Range). The hardware setup was done and the devices were programmed to meet the required application. Communication Performance test was performed in VEBIC and Technobothnia and comparison made to come up with a recommendation of the wireless protocol to implement as the final choice for the Wärtsilä smart NOx sensor case study. The protective casing for the recommended wireless-CAN modules were developed using additive manufacturing technology, precisely the 3D printing.

From the experiments, the XBee had a better RSSI value and security feature over the other wireless modules used. The latency values of the wireless solution has BLE with the best value of maximum latency of 60ms, followed by XBee with maximum latency value of 106ms which is very good based on what it is been applied to. These features and the good performance in the packet loss test, the ability to enhance the battery life, and a better penetrating capability and range when compared to the BLE leads to the recommendation of implementing this wireless protocol as the main choice in the Wärtsilä smart NOx sensor case study. The XBee can be used to create a network of sensors. This implies that several sensors values can be read and transmitted for monitoring on the speedgoat. The XBee network could be made up of several END devices or nodes (sensors) and a COORDINATOR device as the ZigBee protocol supports 3 nodes types namely ZigBee Coordinator ZC, ZigBee Router (ZR) and ZigBee End Device (ZED). The BLE implementation comes up as the second best in respect to the battery life, good RSSI value and security feature. In addition, the good performance in the packet loss test and the ability to enhance the battery life leads to the recommendation of implementing this wireless protocol as an alternative in the Wärtsilä smart NOx sensor case study.

Future work on this could include further research on extending battery life by optimizing all parameters associated with data rates, current and energy consumption can be performed. The research could be based on optimizing the software to improve the battery life of the present prototype. This can also be achieved by integrating Field Programmable Gate Arrays (FPGAs) in the Wireless Sensor Node. The advantage of FPGA is that it allows processing larger amounts of data, with smaller memory footprint by using pipelining to process the data “online” while it is buffered instead of storing it in some off-chip memory. Reducing off-chip memory usage will greatly reduce energy consumption as well. Processing raw data locally in the node will reduce wireless transmission (saving energy) which is desirable in the increasingly crowded unlicensed wireless spectrum. Instead of transmitting the whole raw data, it can be processed in the FPGA for feature extraction, such as frequency analysis, machine vision, fault detection etc. Since most of the power is consumed in wireless transmission, transmitting extracted features only can significantly improve battery life for remotely installed wireless

sensor nodes which might be expensive or not possible to replace their batteries. The research can also be extended to include the ability to implement Over the Air firmware updates. Since FPGAs are Field Programmable Gate Arrays, it is possible to reconfigure the hardware in the field to implement new DSP algorithms, fix bugs or improve performance. By integrating FPGA with a wireless module that supports OTA firmware updates, it is possible to create a fully reconfigurable wireless sensor node, in which both the wireless stack software and the hardware acceleration (FPGA) can be remotely upgraded to improve or add new services. This research can be extended to implement wireless communication with more than one smart NO_x sensor where several sensor values are being monitored.

LIST OF REFERENCES

3dfury (2012). *IEEE 802.15.4 (ZigBee radio) Technology*.

Available online: <http://www.3dfury.eu/dictionary/ieee-802-15-4.html>.

Adafruits Industries (2018). *Introduction to Bluetooth Low Energy*.

Available at: <https://cdn-learn.adafruit.com/downloads/pdf/introduction-to-bluetooth-low-energy.pdf>.

Adame, Toni, Albert Bel, Boris Bellalta, Jaume Barcelo & Miquel Oliver (2014).

IEEE 802.11AH: The WIFI Approach for M2m Communications. IEEE Wireless Communications. December 2014. Available online: <https://ieeexplore.ieee.org/document/7000982>.

Advantech B+B SmartWorx (2018). *Industrial Wireless: Selecting a wireless*

technology. Available at: <http://www.bb-elec.com/Learning-Center/All-White-Papers/Wireless-Cellular/Industrial-Wireless-Selecting-a-Wireless-Technolog.aspx>.

Argenox Technologies (2018). *Introduction to Bluetooth Low Energy (BLE) v4.0*.

Available online: <http://www.argenox.com/bluetooth-low-energy-ble-v4-0-development/library/introduction-to-bluetooth-low-energy-v4-0/>.

Automation (2018). *Industrial Wireless: Solving Wiring Issues by Unplugging*.

Available online: <https://www.automation.com/library/articles-white-papers/wireless-networks-io/industrial-wireless-solving-wiring-issues-by-unplugging>.

Bluetooth SIG (2018a). *Bluetooth*. Available at: <https://www.bluetooth.com/>.

Bluetooth SIG (2018b). *Security, Bluetooth Low Energy*. Available for download:

https://www.google.fi/url?sa=t&rct=j&q=&esrc=s&source=web&cd=8&cad=rja&uact=8&ved=2ahUKEwic9Zmfi_zdAhUFWCwKHRvQCz0QFjAHegQIBBAC&url=https%3A%2F%2Fwww.bluetooth.com%2F~%2Fmedia%2Ffiles%2Fspecification%2Fbluetooth-low-energy-security.ashx&usg=AOvVaw0TEh4kStK-I3CyR8WzYoBy.

Boukerche, Azzedine (2006). *Handbook of algorithms for wireless networking and mobile computing*. Broken Sound Parkway NW. Chapman and Hall/CRC. Available online: https://doc.lagout.org/science/0_Computer%20Science/2_Algorithms/Handbook%20of%20Algorithms%20for%20Wireless%20Networking%20and%20Mobile%20Computing%20%5BBoukerche%202005-11-28%5D.pdf.

Carstens, Stefan & W. Addy Majewski (2018). *NOx Sensors*. Available online: https://www.dieseln.net/tech/sensors_nox.php.

Cooking-Hacks (2018). *cooking-hacks.com* Available online: <https://www.cooking-hacks.com>.

Conley, Bill (2018). *Solving Industrial Monitoring Challenges through Wireless I/O. Industrial Wireless*. White Paper. B&B Electronics. Available at: <http://www.bb-elec.com/Learning-Center/All-White-Papers/Wireless-Cellular/Solving-Industrial-Monitoring-Challenges-through-W.aspx>.

Darshana, Thomas, Edward Wilkie & James Irvine (2016). *Comparison of Power Consumption of WiFi Inbuilt Internet of Things Device with Bluetooth Low Energy*. World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering Vol:10, No:10, 2016. Available at: <https://waset.org/publications/10005628/comparison-of-power-consumption-of-WiFi-inbuilt-internet-of-things-device-with-bluetooth-low-energy> and

<https://pdfs.semanticscholar.org/73cb/7616c46a37df9db3783a51ddee4a0599856d.pdf>.

Department of Communications, Climate Action and Environment (2018).

EU Clean Air Policy. Available online: <https://www.dccae.gov.ie/en-ie/environment/topics/air-quality/eu-clean-air-policy/Pages/default.aspx>.

DIGI (2016). *Wireless Connectivity Kit Getting Started Guide*. Available online:

<https://www.digi.com/resources/documentation/digidocs/pdfs/90001456-13.pdf>.

DIGI (2017). *Signal strength and the RSSI pin*. Available online:

https://www.digi.com/resources/documentation/Digidocs/90001456-13/Default.htm#concepts/c_rssi_pin_and_signal_strength.htm%3FTocPath%3DSig-nal%2520strength%2520and%2520radio%2520frequency%2520range%7C_____3.

DIGI (2018). *XBee®/XBee-PRO S2C Zigbee® RF Module User Guide 2018*.

Available online: <https://www.digi.com/resources/documentation/digidocs/pdfs/90002002.pdf>.

Digi-Key (2018). *Optimizing Security Sensor Battery Life*.

A contribution by Ember Corporation. Available at: <https://www.digikey.com/en/articles/techzone/2011/mar/optimizing-security-sensor-battery-life>.

Elmusrati, Mohammed (2017). *Advanced Telecommunication Theory*.

Course Material at the University of Vaasa.

ES Electronics-Shop (2018). *XBee Explorer USB*. Available online:

<https://www.electronic-shop.lu/EN/products/153968>.

- European Environment Agency (EEA) (2018). *Nitrogen oxides (NOx) emissions*. Available online: <https://www.eea.europa.eu/data-and-maps/indicators/eea-32-nitrogen-oxides-nox-emissions-1>.
- Frolic, Kai (2016). *Fresnel zone*. Available online: <https://www.pagerpower.com/news/fresnel-zone/>.
- Gao, Xiang, Dagui Huang, Yuanqiang Chen, Wei Jin & Yi Luo (2013). *The Design of a Distributed Control System Based on CAN bus*. Proceedings of 2013 IEEE, International Conference on Mechatronic and Automation, August 4 – 7, Takamatsu, Japan. Available online: <https://zapdf.com/the-design-of-a-distributed-control-system-based-on-can-bus.html>.
- GE Additive (2018). *What is Additive manufacturing?* Available online: <https://www.ge.com/additive/additive-manufacturing>.
- Github (2018). *CAN-on-DSP-TMS320F28335 wiki*. Available online: <https://github.com/zhanglongqi/CAN-on-DSP-TMS320F28335/wiki>.
- Goursaud C. & Gorce J.M. (2015). *Dedicated networks for IoT: PHY / MAC state of the art and challenges*. EAI endorsed transactions on Internet of Things.
- Gstatic (2018). *NOx sensor pins*. Available online: https://encryptedtbn0.gstatic.com/images?q=tbn:ANd9GcSrF2I9nmI9c5E1IxU4SWd7UmXJNWFqHrLaywFzRHiAB_53a_bmCw.
- Ina, Senft & Lemire Bertrand (2010). *Specification Smart NOx Sensor "Uninox24V"*. Provided by Wärtsilä.
- Khan, Ateeq, & Klaus Turowski. (2016). *A survey of current challenges in manufacturing industry and preparation for industry 4.0. Advances in Intelligent Systems and Computing*. pp. 15-26. Available online:

https://www.springer.com/cda/content/document/cda_downloaddocument/9783319336084-c2.pdf?SGWID=0-0-45-1564143-p179960859.

Kvaser (2018a). *CAN Messages*. Available at:

<https://www.kvaser.com/about-can/the-can-protocol/can-messages-13/>.

Kvaser (2018b). *CAN Messages*. Available online:

<https://www.kvaser.com/about-can/the-can-protocol/can-messages-23/>.

Lafenergy (2017). *Power and Energy*. Available online:

<http://lafenergy.org/essays/energy.php>.

Lethaby, Nick (2017). *Wireless connectivity for the Internet of Things*.

Texas Instrument (2017). Available online:

<http://www.ti.com/lit/wp/swry010a/swry010a.pdf>.

Libelium Comunicaciones Distribuidas S.L (2017a). *Waspote 802.15.4 Networking Guide*. Available at:

http://www.libelium.com/downloads/documentation/waspote-802.15.4-networking_guide.pdf.

Libelium Comunicaciones Distribuidas S.L (2017b). *LoRa Networking Guide*.

Available online:

http://www.libelium.com/downloads/documentation/waspote_lora_868mhz_915mhz_sx1272_networking_guide.pdf.

Libelium Comunicaciones Distribuidas S.L (2017c). *WiFi-PRO Networking Guide*.

Available at:

http://www.libelium.com/downloads/documentation/wifi_networking_guide.pdf

Libelium Comunicaciones Distribuidas S.L (2017d). *Bluetooth Low Energy Networking Guide*.

Available at: http://www.libelium.com/downloads/documentation/bluetooth-low-energy-networking_guide.pdf.

Libelium Comunicaciones Distribuidas (2018a). *Waspote datasheet v7.7*. Available at: http://www.libelium.com/downloads/documentation/waspote_datasheet.pdf.

Libelium Comunicaciones Distribuidas (2018b). *Waspote Technical Guide*.

Available at: http://www.libelium.com/downloads/documentation/waspote_technical_guide.pdf.

Libelium Communication Distribution (2018c). *Extreme range lora-sx1272 module*.

Available at: <https://www.cooking-hacks.com/documentation/tutorials/extreme-range-lora-sx1272-module-shield-arduino-raspberry-pi-intel-galileo>.

Lin, Robert (2014). *SSL/ TLS Cipher Suite Analysis and strong Cipher Enablement*.

Available online: https://community.digicert.com/content/usergenerated/asi/rdbms/attachments/sites/connect/en/blogs/_jcr_content/content/primary/blog/ssl-ciphers-beyond-private-key-and-certificate/Ciphersuite%20analysis%20and%20strong%20cipher%20enablement.pdf.

LoRa Alliance (2015). *A Technical Overview of LoRa and LoRaWAN*.

Available online: https://www.tuv.com/media/corporate/products_1/electronic_components_and_1asers/TUeV_Rheinland_Overview_LoRa_and_LoRaWANtmp.pdf.

MikroElektronika (2018). CAN SPI 3.3V click schematics. Available online:

https://libstock.mikroe.com/img/projects/21707/245/1326215999_can-spi_thumb.jpg.

- Miller, Robert (2016). *LoRa Security, Building a Secure LoRa Solution*. MWR Labs Whitepaper. PP 7-8. Available at: <https://labs.mwrinfosecurity.com/assets/BlogFiles/mwri-LoRa-security-guide-1.2-2016-03-22.pdf>.
- Mukherji, Arup & Subbanarasaiah Sadu (2016). *ZigBee Performance Analysis*. Proceeding from IEEE Wireless Communications, Signal Processing and Networking (WiSPNET) 2016 conference. Pp 328 -329.
- Nilsson, Staffan (2018). *Controller Area Network - CAN Information*. Available at: <http://hem.bredband.net/stafni/developer/CAN.htm>.
- Oath Tech Network Aol Tech (2018). *Bluetooth SIG unveils Smart Marks, explains v4.0 compatibility with unnecessary complexity*. Available at: <https://www.engadget.com/2011/10/25/bluetooth-sig-unveils-smart-marks-explains-v4-0-compatibility-w/>.
- Parkers (2018). *What is AdBlue and why does your diesel car have it?* Available online: <https://www.parkers.co.uk/car-advice/2016/what-is-adblue-and-what-does-it-do-for-your-car/>.
- PCtech24 (2017). *Types of network attacks*. Available online: <https://www.pctech24.com.au/blog/types-of-security-threats-and-network-attacks-and-their-counter-measures>.
- ResearchGate (2018). *LoRa Network Architecture*. Available online: https://www.researchgate.net/figure/LoRa-network-architecture_fig1_307965130.
- Rizzi, Mattia*, Paolo Ferrari*, Alessandra Flammini*, Emiliano Sisinni* & Mikael Gidlun† (2017). *Using LoRa for industrial wireless networks*. *Dept. of Information Engineering, University of Brescia Brescia, Italy and †Department of Information Systems and Technology, MidSweden University, Sundsvall, Sweden.

IEEE 2017. Available online:
https://www.researchgate.net/publication/318751577_Using_LoRa_for_industrial_wireless_networks.

Roundy S., D. Steingart, L. Fr chet, P. Wright & J. Rabaey (2004). *Power Sources for Wireless Sensor Networks*, Vol. 2920, pp. 1-17.

Semtech (2017). *SX1272/73 LoRa Datasheet, Rev. 3.1*. Available at:
<https://www.semtech.com/uploads/documents/sx1272.pdf>. (accessed on 19 June 2018.)

SICK 2018. CEMS solutions MCS100E HW. Available online:
<https://www.sick.com/fi/en/analyzer-solutions/cems-solutions/mcs100e-hw/c/g285463>.

Sparkfun (2018). *What is an Arduino?* Available online:
<https://learn.sparkfun.com/tutorials/what-is-an-arduino>.

Speedgoat GmbH (2007-18). *Applications & Industries*. Available at:
<https://www.speedgoat.com/applications-industries>.

Storm, Xiaoguo (2017). *Designer of the Simulink model for receiving Smart NOx CAN frames*. University of Vaasa.

Symmetry Electronics (2018). *WiFi Standards 802.11a/b/g/n vs. 802.11ac: Which is Best?* Available online:
<https://www.semiconductorstore.com/blog/2014/WiFi-standards-802-11a-b-g-n-vs-802-11ac-Which-is-Best/806/>.

Texas Instrument (2016). *Introduction to the Controller Area Network (CAN)*. SLOA101B–August 2002–Revised May 2016. Available at:
<http://www.ti.com/lit/an/sloa101b/sloa101b.pdf>.

- Texas Instrument (2013). *ZigBee Wireless Networking Overview*. Available at:
<http://www.ti.com/lit/sg/slyb134d/slyb134d.pdf>.
- Toni Adame, Albert Bel, Boris Bellalta, Jaume Barcelo & Miquel Oliver (2014).
IEEE 802.11AH: The WiFi Approach for M2m Communications. IEEE Wireless Communications, December 2014. Available online:
https://www.researchgate.net/publication/260268761_IEEE_80211ah_the_WiFi_approach_for_M2M_communications.
- Tutorialspoint (2018). *Arduino - Program Structure*. Available online:
https://www.tutorialspoint.com/arduino/arduino_program_structure.htm.
- United States Environmental Protection Agency (EPA) (2018).
Nitrogen Oxides (NOx) Control Regulations. Available online:
<https://www3.epa.gov/region1/airquality/nox.html#ract>.
- Vançin, Sercan & Ebubekir Erdem (2015). *Design and Simulation of Wireless Sensor Network Topologies Using the ZigBee Standard*. International Journal of Computer Networks and Applications (IJCNA) Volume 2, Issue 3. Pp 135 -137.
 Available online: <http://www.ijcna.org/Manuscripts/Volume-2/Issue-3/Vol-2-issue-3-M-03.pdf>.
- VSkills (2018). *Network attacks*. Available online:
<https://www.vskills.in/certification/tutorial/wimax-4g-2/network-attacks/>.
- Walia, Navjot Kaur, Parul Kalra & Deepti Mehrotra (2016). *An IOT by Information Retrieval approach: Smart Lights controlled using WiFi*. 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence). Available at:
<https://ieeexplore-ieee-org.proxy.uwasa.fi/stamp/stamp.jsp?tp=&arnumber=7508211>.
- Wan, Xiao-feng, Yi-si Xing & Li-xiang Cai (2009). *Application and Implementation of*

CAN Bus Technology in Industry Real-time Data Communication. Proceedings of 2009 IEEE, International Conference on Industrial Mechatronics and Automation (ICIMA). PP 278 -279.

Wikipedia (2018a). *Bluetooth*. Available at:

https://en.wikipedia.org/wiki/Bluetooth#Specifications_and_features.

Wikipedia (2018b). *Electronic Diesel Control*. Available at:

https://en.wikipedia.org/wiki/Electronic_Diesel_Control.

Wikipedia (2018c). *Engine Control Unit*. Available at:

https://en.wikipedia.org/wiki/Engine_control_unit#Sensors_and_actuators.

Wikipedia (2018d). *Bit error rate*. Available at:

https://en.wikipedia.org/wiki/Bit_error_rate.

Wikipedia (2018e). *Network delay*. Available at:

https://en.wikipedia.org/wiki/Network_delay.

Wikipedia (2018f). *Packet loss*. Available at:

https://en.wikipedia.org/wiki/Packet_loss.

Wikipedia (2018g). *Power consumption*. Available at:

https://simple.wikipedia.org/wiki/Power_consumption.

ZigBee Alliance (2012). *ZigBee Specification*. Available at:

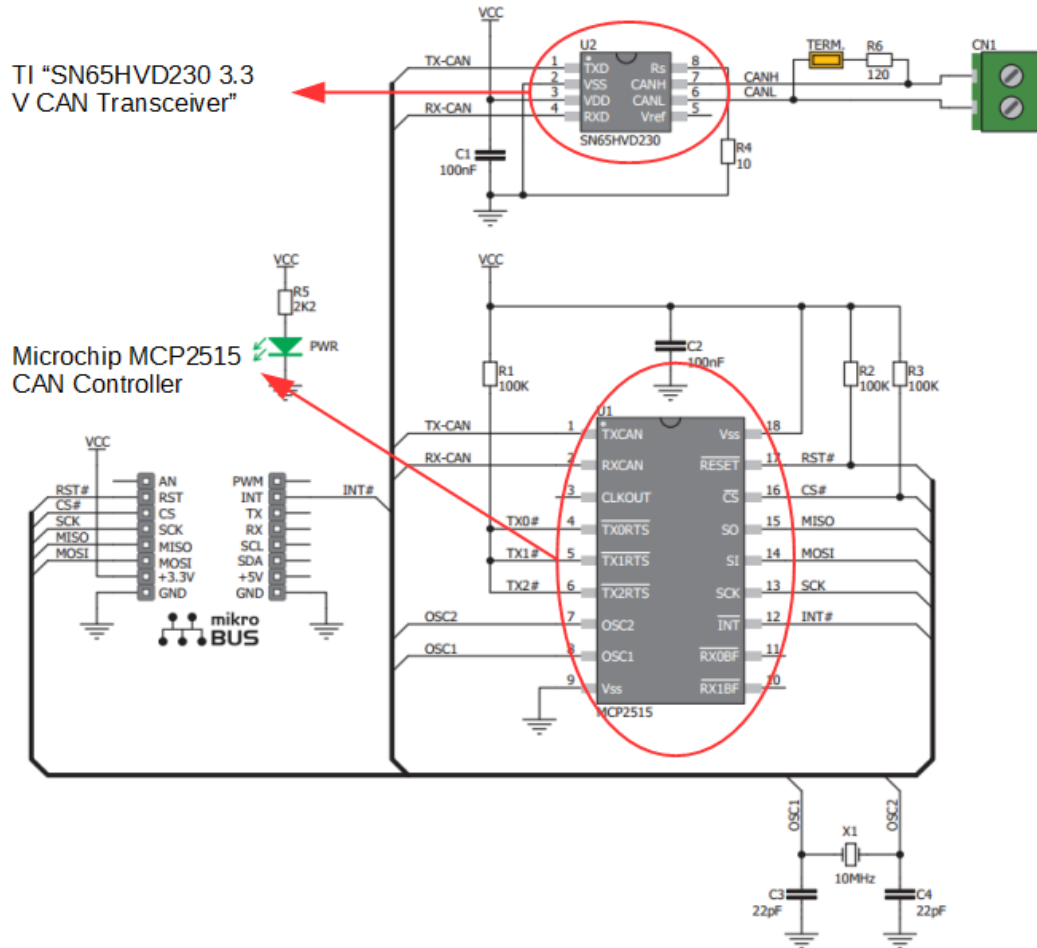
<http://www.zigbee.org/download/standards-zigbee-specification/>.

Zybuluo (2018). *Zigbee Packet Structure*. Available online:

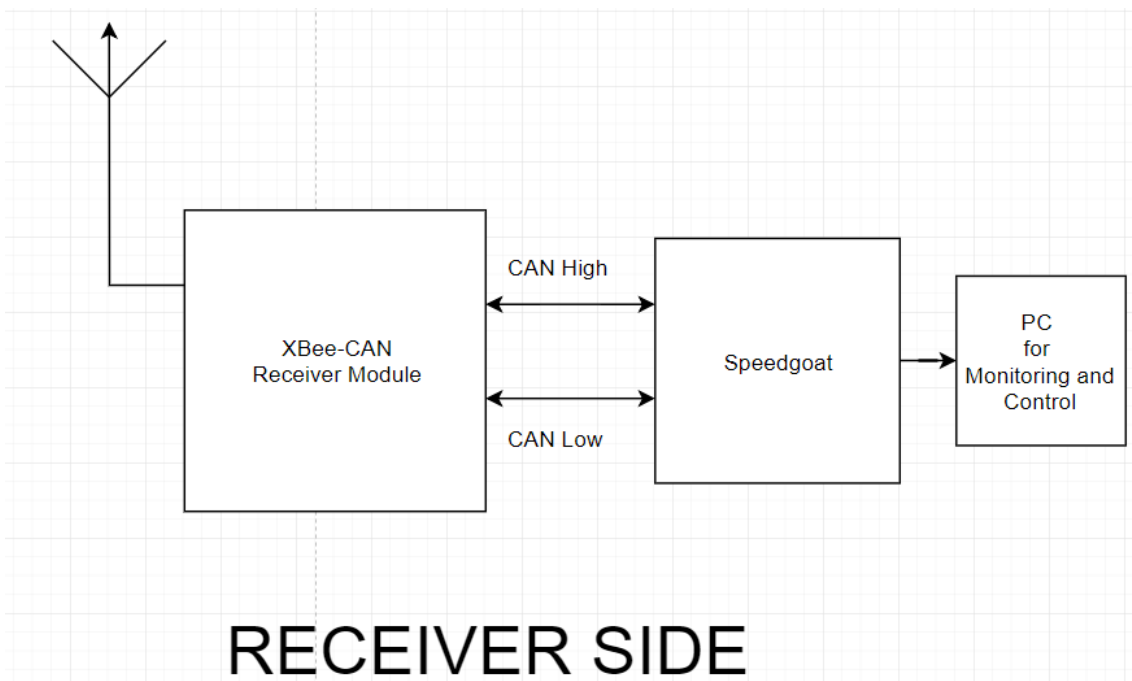
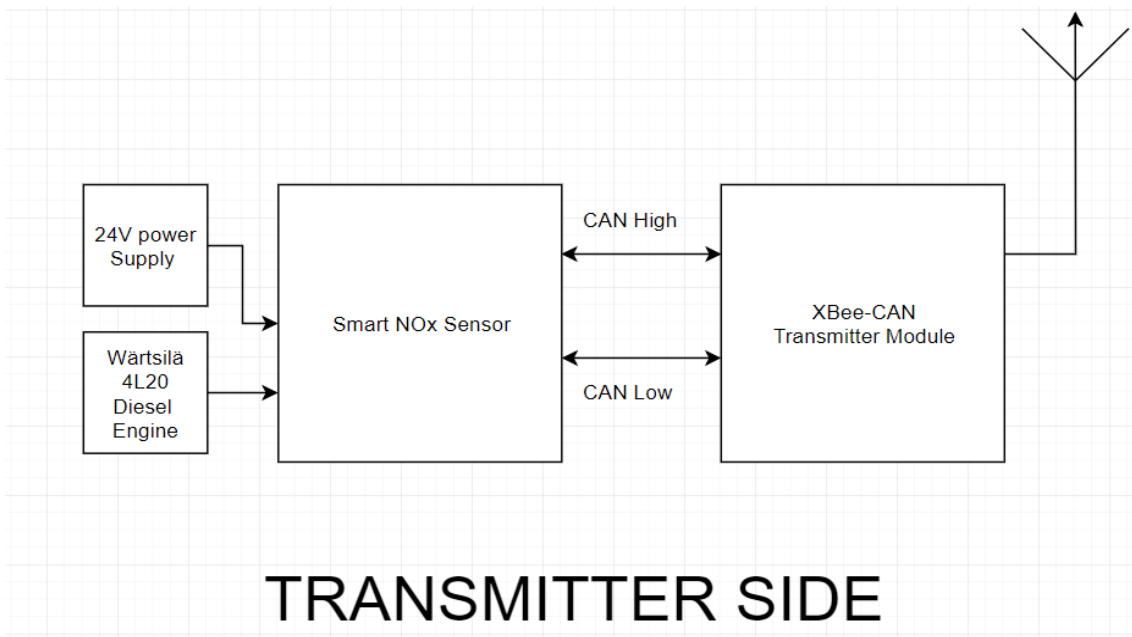
<https://www.zybuluo.com/yiltoncent/note/128986#zigbee-packet-structure>.

APPENDICES

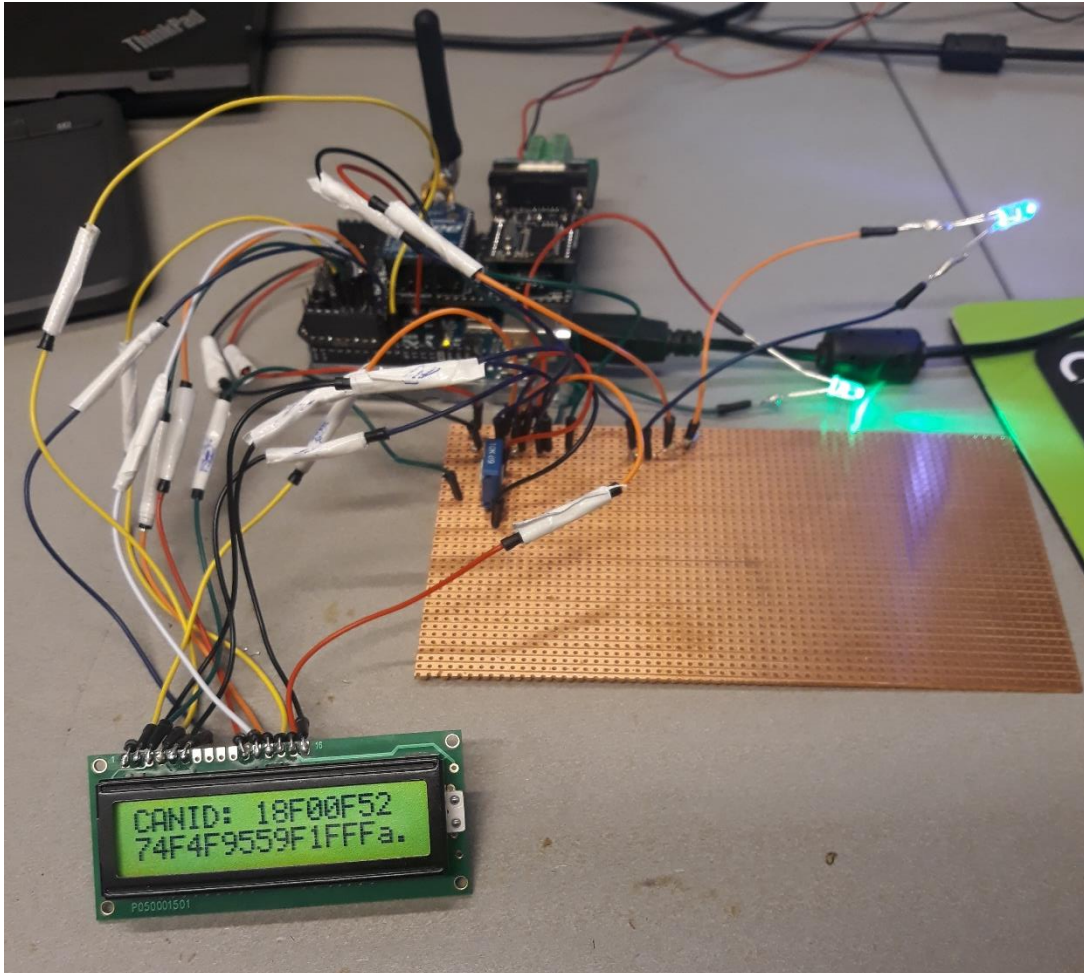
APPENDIX 1. Schematic of Mikroelektronika CAN SPI click board
(MikroElektronika 2018).



APPENDIX 2. Smart NOx, XBee-CAN Module and Speedgoat system overview



APPENDIX 3. LCD Display for Transmitter/Receiver Modules



APPENDIX 4. Sample output of transmitter and receiver code

Sample output of receiver code.

```
-----
-> data from XBee transmitter
0x71 0x9E 0xCF 0x61 0xF2 0x61 0x69 0x4D 0x48 0x62 0xD3 0xC5 0xC2 0x39 0xBD
0xA9
-> RSSI Value: 47

Decryption started 128 bits

KEY = 48656c6c6f776f726c64796f75726f63
CIPHERTEXT = 719ecf61f261694d4862d3c5c239bda9
0x71 0x9E 0xCF 0x61 0xF2 0x61 0x69 0x4D 0x48 0x62 0xD3 0xC5 0xC2 0x39 0xBD
0xA9
Decryption Completed
PLAINTEXT = 22a60f52f8559f1fff1108a60f45f855

-> Decrypted Data from XBee Transmitter:
ErrorDectioNumber[10] + Smart NOx Data[1]to[8] + checksum[9] + PaddingDa-
ta[10]to[15]
0x22 0xA6 0xF 0x52 0xF8 0x55 0x9F 0x1F 0xFF 0x11 0x8 0xA6 0xF 0x45 0xF8 0x55

SumofSmartNOxDATAFrom_XBeeTransmitter is: 411

Data ErrorDectioNumber is: 22

checksumData is: 11
Receiver Error Detection Number is: 422

-> data to Speedgoat
CAN ID: 0x18F00F52
0xA6 0xF 0x52 0xF8 0x55 0x9F 0x1F 0xFF
```

Sample output of transmitter code.

```
-----
-> ErrorDectNum[1] + Smart NOx Data[8] + checksum[1]:
0x22 0xA6 0xF 0x52 0xF8 0x55 0x9F 0x1F 0xFF 0x11

SumofDataFrom_smartNOx: 411

Data ErrorDectioNumber is: 22

checksumData is: 11
Sender Error Detection Number is: 422

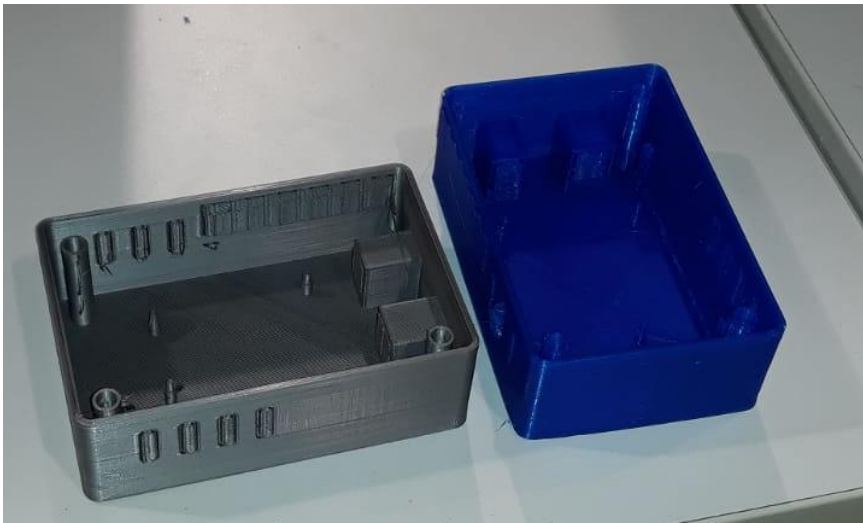
Encryption started 128 bits

KEY = 48656c6c6f776f726c64796f75726f63
PLAINTEXT = 22a60f52f8559f1fff1108a60f45f855
0x22 0xA6 0xF 0x52 0xF8 0x55 0x9F 0x1F 0xFF 0x11 0x8 0xA6 0xF 0x45 0xF8 0x55
Encryption Completed
CIPHERTEXT = 719ecf61f261694d4862d3c5c239bda9

-> Encrypted Data for XBee Transmission:
0x71 0x9E 0xCF 0x61 0xF2 0x61 0x69 0x4D 0x48 0x62 0xD3 0xC5 0xC2 0x39 0xBD
0xA9

XBee Sending data...
```

APPENDIX 5. 3D printed protective casing body



APPENDIX 6. 3D printed protective casing covers

