UNIVERSITY OF VAASA

FACULTY OF TECHNOLOGY

TELECOMMUNICATION ENGINEERING

Maiwulan Wulayinjiang

WIRELESS DATA LOGGER - A JOINT USE OF FREQUENCY CONVERTER AND WIRELESS SENSOR NETWORK

Master's thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, June 23, 2014.

Supervisor	Professor Mohammed Salem Elmusrati
Instructor	M.Sc. Reino Virrankoski

ACKNOWLEDGEMENT

I would like to thank Professor Mohammed Elmusrati and Senior Researcher Reino Virrankoski for their excellent guidance during this thesis work.

I am very grateful for Markus Madetoja and Linh Le Manh for their great support, which led to accomplish this project on time.

I am also thankful to Juhani Kerovuori for his great hospitality and support during factory tests in Riihimäki.

TABLE OF CONTENTS **Page**

ABBREVIATIONS

ACL	Access Control List
API	Application Programming Interface
BER	Bit Error Rate
BSD	Berkeley Software Distribution
Bufbuffer	Buffer of Buffers
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
FFD	Full Function Device
FIFO	First In First Out
GK	Group Keying
GTUI	Graphical Test User Interface
HDMI	High Definition Multimedia Interface
HSC	Hardware Stack Connector
IP	Internet Protocol
ISM	Industrial Scientific Medical
ISR	Interrupt Service Register
LR-WPAN	Low Rate Wireless Personal Area Network
MAC	Medium Access Control
MC	Main Controller
MOS	MANTIS Operating System
NSK	Network Shared Keying
P2P	Peer to Peer
PAN	Personal Area Network
PHY	Physical Layer

PK	Pairwise Keying
QoS	Quality of Service
RFC	Radio Frequency Controller
RFD	Reduced Function Device
RFID	Radio Frequency Identification Technology
RRC	Response Request Command
RTOS	Real Time Operating System
TCP	Transmission Control Protocol
UART	Universal Synchronous and Asynchronous Receiver/Transmitter
UI	User Interface
USB	Universal Serial Bus
UWB	Ultra Wideband
VHMI	Vacon Human Machine Interface
WLAN	Wireless Local Area Network
WSN	Wireless Sensor Network
ZC	ZigBee Coordinator
ZED	ZigBee End Device
ZR	ZigBee Router

UNIVERSITY OF VAASA

Faculty of Technology

Author:	Maiwulan Wulayinjiang
Topic of the Thesis:	Wireless Data Logger – A Joint Use of Frequency Converter and Wireless Sensor Network
Supervisor:	Professor Mohammed Salem Elmusrati
Instructor:	Reino Virrankoski
Degree:	Master of Science in Technology
Department:	Department of Computer Science
Degree Programme:	Master's in Telecommunication Engineering
Major of Subject:	Telecommunication Engineering
Year of Entering the University:	2012
Year of Completing the Thesis:	2014 **Pages:** 71

ABSTRACT

"Smart Industry" is a new unavoidable trend in vast varieties of industry fields. In the case of developing smart crane systems, cutting edge innovation and design is required. Many crane manufactures have expressed their strong interest in applying wireless technology to their crane products. Recent research achievements in wireless sensor node development have created technologically mature, cost effective solutions for many applications. When either monitoring or controlling the crane, one must have access to the frequency converter first. As for the purpose of analyzing the behavior of crane, the Wireless Sensor Network can be used to collect data from frequency converters.

In this thesis, a wireless sensor network system was designed and developed in order to collect data from several frequency converters. The UWASA Node, a wireless sensor node designed by researchers from Aalto University and University of Vaasa, was implemented for establishing this wireless data logging network. As a result, the system has an ability of logging continuous data as well as the changes of data in user defined logging interval. Additionally, the reliability of the wireless transmission was investigated and possible solutions were presented.

KEYWORDS: Frequency Converter, UWASA Node, WSN, Wireless Data Logger

1. INTRODUCTION

Wireless Automation has been a hot research topic in recent years. The mobility, flexibility and cost effective feature of wireless communication caught the attention of academic researchers as well as industries. The fast development of various wireless communication technologies in recent years has made it possible to monitor the industrial system behaviors in a wireless manner.

Wireless sensor networks can be utilized for industrial automation, healthcare, military, agriculture and so on. The researchers from Aalto University and University of Vaasa have developed a wireless sensor node called the UWASA Node, which is especially targeted for wireless automation applications. The UWASA Node is a generic, modular and stackable wireless sensor platform (Yiğitler, Virrankoski & Elmusrati: 1). It can be reprogrammed according to the need of the application.

In crane manufacturing industries, there are a certain amount of frequency converters that are deployed to control the electric motors. In the physical memory of a frequency converter, there are many parameters that indicate the status of motor, for example frequency, voltage and so on. Once the values of these parameters are known, they can be used for monitoring and analyzing the performance of the crane system. Therefore, collecting selected parameters' values from frequency converters with Wireless Sensor Network (WSN) has become the motivation of this research.

The main goal of this thesis work was to design and build a wireless sensor network by using seven UWASA Nodes in order to collect certain parameters' data from frequency converters. The collected data could be saved in a database and then further transmitted over Ethernet to the user's end device (Laptop or Tablet) for analyzing the system behavior. The overall structure of this wireless data logging system is illustrated in **Figure 1,** and the main focus of this particular thesis is represented by the left hand side of the figure.

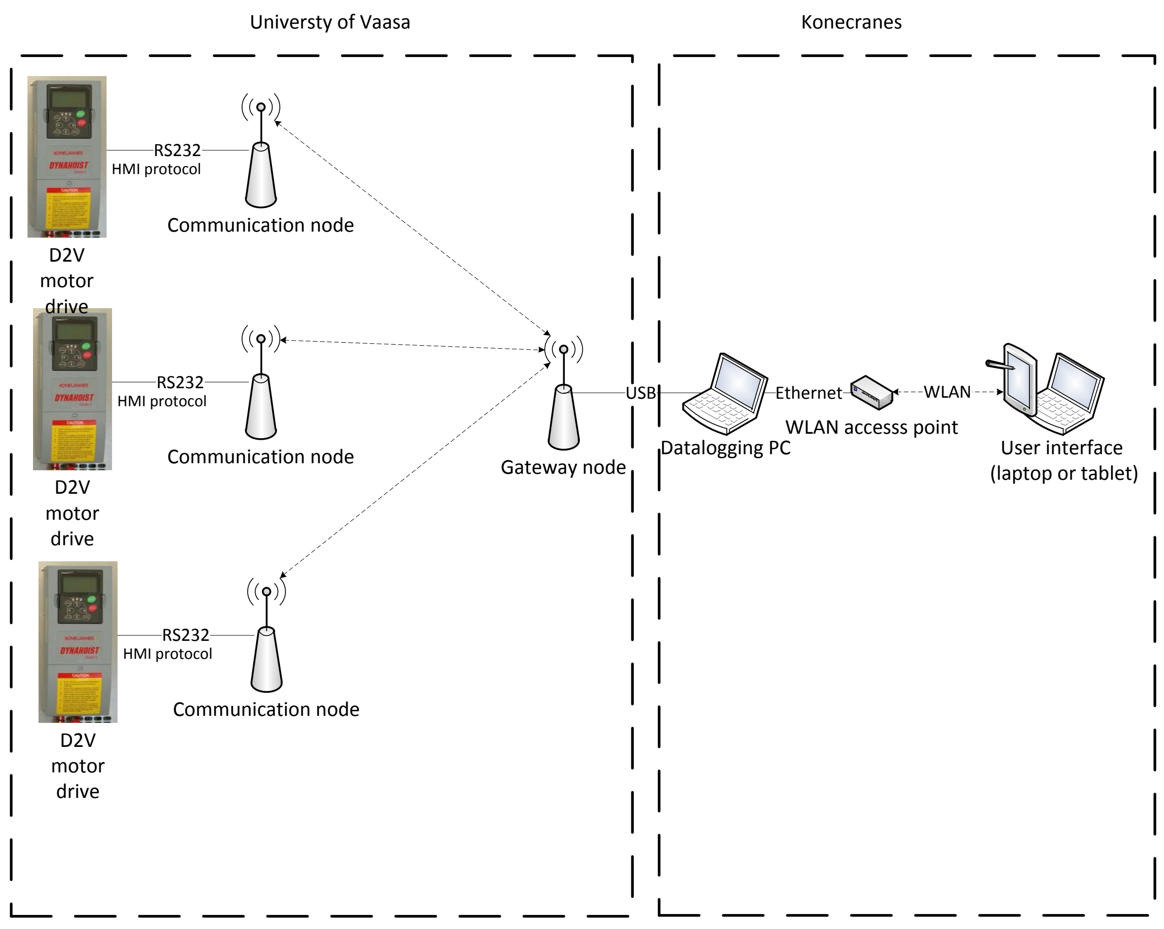


Figure 1. Overall structure of the Wireless data logger system for frequency converters (Kerovuori & Virrankoski 2013: 2).

This thesis consists of seven chapters. Concepts of Smart Industry and Wireless Automation are presented in the first chapter. The most relevant background theories to Wireless Sensor Networks are introduced in Chapter 2. The core part of the thesis is located in Chapter 4 and 5 which describe the hardware and software architecture of the wireless data logging system in detail. The experiments and results are presented in Chapter 6. Finally, the Chapter 7 summarizes the whole thesis, and gives a quick look to the future work.

2. WIRELESS AUTOMATION

Smart devices are changing our daily life. The ability of wireless communication is one of the key factors to achieve smart-functionality and mobility in smart devices. However, in the world of “Smart Industry” development, the situation is more complicated. Because of the limitations in the computation, communication and energy supply of the wireless sensor nodes and actuators, it is challenging to use them in industrial automation. In this chapter, a brief overview of the concepts in wireless automation is introduced in four sections.

2.1. Opportunities

Currently, the implementation of wireless technology in automation systems is significant. In the case of monitoring industrial plants, the WSN is considered an attractive solution in such cases where the place is not reachable by cabled systems. Wireless sensor nodes are capable of sensing critical elements in a cost effective way. The collected sensor data can give indication of process problems. The efficiency of system monitoring processes is achieved by wireless communication between sensor nodes. Thus, the responsiveness of the system to process problems can be improved.

There are also many potential applications that can be targeted to increase the safety of industrial systems, for example fire and gas detection and elimination of spills. The use of wireless technology can help increase safety by having fewer blind spots around the industrial plants, hence resulting in improved safety on site (Waqas Ikram & Nina F. Thornhill, 2010).

All automation systems tend to be cost effective. For this reason, Wireless Automation solutions can reduce the use of wired systems, which can improve the cost efficiency of the overall system. Moreover, some harsh industrial environments would not allow the deployment of wired networks. In such cases, wireless technology turned out to be the only possible solution.

2.2. Requirements

In every automation system or industrial plant, communication network is used to exchange information between sensors, actuators, machines, control and monitoring devices. Before applying wireless technology, one must assure that the technology can fulfill the particular system requirements. In this section, the requirements that are essential to achieve a wireless automation system are introduced in two different perspectives: basic requirements and wireless associated requirements.

2.2.1. Basic Requirements

The hierarchy of an industrial system is illustrated in **Figure 2**. It is characterized into three levels. The Field Level contains the actual production processes. The Cell Level comprises of all control processes of one production line whereas Factory Level takes care of all business process (Kaleem, Stefan & Uwe 2008: 6).

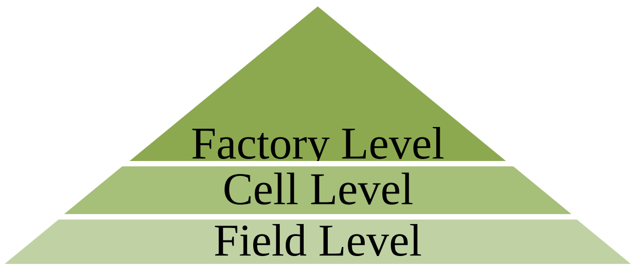


Figure 2. The process hierarchy of industrial systems (Kaleem et al 2008: 6).

The basic requirements of industrial automation systems are introduced according to the hierarchy division (see **Table 1**). As it is shown in **Table 1**, different hierarchy levels of industrial systems have different requirements, thus specific sectors in each level should be designed individually.

The data handling and real-time transmission are the most critical aspects for any industrial system. Therefore, the wireless automation network should be able to handle different types of data transmission precisely. For example, the periodic data is mostly linked to inputs and outputs of control algorithms, and it is required to transmit within a certain amount of time (Decotignie 2002). In this case, the wireless network transmission delay must be less than the maximum latency, which is allowed in the system specifications.

Table 1. Basic requirements of industrial automation systems according to hierarchical division (Kaleem et al 2008: 7).

Requirement	**Factory Level**	**Cell Level**	**Field Level**
Availability	Low to medium	High	High to very high
Typical data traffic	Acyclic	Cyclic and acyclic	Mainly cyclic, acyclic
Safety	Medium to high	High	High to very high
Real-time data transmission	No	Yes	Yes
Typical data length	Long	Medium	Short
Bandwidth	Broadband	Medium	Barrow/broadband
Number of stations	High to very high	Medium	Medium to high
Type of station	Stationary and mobile peripherals	Stationary and mobile peripherals	Stationary and mobile devices
Distance	Long	Medium	Short to medium

2.2.2. Wireless Associated Requirements

Mobility is a great advantage in wireless systems. However, to make the best of it, the mobility of a wireless automation system needs to meet certain requirements. The wireless network should offer connectivity to both static and mobile nodes in industrial plants. The wireless devices that are deployed onto fast moving components of automation system should be able to overcome Doppler Effect. In addition to that, the data transmission range of wireless sensor nodes should be large enough to satisfy the mobility requirement of a specific automation system.

Security of wireless communication has been one of the important issues since the information is transmitted over open air. According to Sikora (2007), the wireless automation systems should be designed in such a way that the information is encrypted and protected against eavesdropping as it may contain vital information about the workflow. Furthermore, malicious packets should be discarded (Persson 2007).

The energy efficiency also is a key factor for a successful industrial plant. From this point of view, any wireless devices that are used for wireless automation systems should have very small energy consumption. For battery operated wireless devices like sensors, the power consumption should be as low as possible. In fact, it is one of the main challenges in WSN development that one must achieve as high performance as possible with the lowest possible energy consumption.

Last but not least; high reliability is a very important requirement in any automation system. Wireless automation systems are required to provide reliable data transmission even in multipath and motion based industrial environments. The communication reliability indicated by the level of connection loss, and packet loss must be kept as low as possible. The average data transmission delay as well as its variation should be kept within a certain acceptable level. The co-channel interface and adjacent-channel interface are common in every wireless communication system. Therefore, one of the main targets in the network design must be the minimization of the disturbing interface. Suitable power management methods need to be used to increase the reliability of wireless data transmission. Additionally, a self-troubleshooting functionality should be implemented in case of unpredictable software upset during the communication system runtime.

2.3. Overview of Radio Technologies

Some of the radio technologies commonly applied for wireless automation are briefly presented in this section.

2.3.1. Wi-Fi

Wireless Local Area Network (WLAN) was first introduced in 1997, and was subsequently branded as Wi-Fi for marketing purposes. It has been standardized within IEEE 802.11, and has bandwidth range of 2-7000 Mb/s. According to Hoefel (2014), the 802.11ac amendment approved by the end of 2013 stated that the maximum data rate can be 7 Gb/s. All different Wi-Fi standards are compared in **Table 2**.

Wi-Fi could be used for higher bandwidth requirement applications; however it has relatively high energy consumption and complex hardware structure. The use cases of Wi-Fi for wireless automation include surveillance cameras, machine vision and other quality control tasks. Wi-Fi is a good solution for the applications where a high bandwidth is needed.

Table 2. IEEE 802.11 standards comparison.

Standard	Release	Freq. (GHz)	Max data rate (Mbps)	Indoor range
802.11	1997	2.4	2	-
802.11 a	Sep 1999	5	54	0 ~ 35
802.11 b	Sep 1999	2.4	11	0 ~ 35
802.11 g	Jun 2003	2.4	54	0 ~ 35
802.11 n	Oct 2009	2.4/5	500	0 ~ 70
802.11 ac	Dec 2013	5	7000	0 ~ 50

2.3.2. ZigBee

ZigBee was introduced in 2002 by ZigBee Alliance. ZigBee is designed to address the unique needs of low-cost, low-power wireless sensor and control networks. It is based on IEEE 802.15.4 standard, and operates at 2.4 GHz radio frequency (ZigBee Alliance 2014).

There are three types of devices in ZigBee architecture: ZigBee Coordinators, ZigBee Routers and ZigBee End Devices. The ZigBee Coordinator (ZC) is at the core of the ZigBee network. It has duties such as controlling the formation and the security of networks. The ZigBee Router (ZR) extends the range of network by connecting ZC and ZDE, and routes the messages inside the wireless network. The ZigBee End Device (ZED) performs specific sensing or control functions in the network. Sensor nodes operating in a ZigBee network are usually low power boards and able to work for years on battery (ZigBee Alliance, Document 053474r17, 2008).

ZigBee takes full advantage of a powerful IEEE 802.15.4 physical radio standard and operation in unlicensed bands worldwide at 2.4GHz (global), 915 MHz (Americas) and 868 MHz (Europe). Raw data throughput rates of 250Kb/s can be achieved at 2.4GHz (16 channels), 40Kb/s at 915 MHz (10 channels) and 20Kb/s at 868 MHz (1 channel). Transmission distances range from 10 to 100 meters, depending on power output and environmental characteristics (ZigBee Alliance 2014).

According to ZigBee Alliance (2014), ZigBee technology is widely used in various control tasks, for example Building Automation, Remote Control, Smart Energy, Retail Services and Telecom Services.

2.3.3. Bluetooth

Bluetooth was created by Ericsson in 1994. It operates within the Industrial, Scientific and Medical (ISM) band at 2.4 to 2.485 GHz radio frequency. The physical and medium access control layers of Bluetooth have been standardized within IEEE 802.15.1, and it provides a low power, low complexity wireless network compared to Wi-Fi, but with lower data rate (see **Table 3**) (Jantunen 2008: 27-28).

Bluetooth standard specifies three device classes with different maximum transmit power level and transmission range as shown in **Table 4**. There are many applications using Bluetooth, the most typical one is a wireless audio input/output interface (headset). According to Jantunen (2008), Bluetooth can be useful in some wireless automation applications because of the wide availability of cheap components. It also provides the possibility of using standard mobile phones or personal computers as control devices.

Table 3. Bluetooth versions and maximum data rate.

Bluetooth Version	Max Data Rate
v1.0, v1.0B, v1.1	768 kb/s
V1.2	1 Mb/s
V2.0, v2.1 (+EDR)	3 Mb/s
V3.0 + HS, v4.0, v4.1	24 Mb/s

Table 4. Bluetooth power levels and range (Jantunen 2008: 28).

Class	Max transmit power/mW	Range
1	100 mW	0 ~ 100 m
2	2.5 mW	0 ~ 10 m
3	1 mW	0 ~ 1 m

2.3.4. RFID

RFID refers to Radio Frequency Identification Technology. The main advantage of this technology is that the sensor is powered by the reading signal, which eliminates the need of separate power source and enables a smaller size (smaller than Bluetooth and ZigBee). On the other hand, the RFID reader device requires much more power than usual sensor nodes and its size is also bigger. The possible frequency bands and data transmission ranges are shown in **Table 5**.

Table 5. The frequency bands and transmission ranges of RFID (Sen et al. 2009).

Band	**Regulations**	**Range**	**Data speed**
120 - 150 KHz	Unregulated	10 cm	Low
13.56 MHz	ISM band	10 cm ~ 1m	Low to moderate
433 MHz	Short Range Devices	100 m	Moderate
865 - 868 MHz (Europe) 902 - 928 MHz (North America)	ISM band	12 m	Moderate to high
2450 - 5800 MHz	ISM band	1 ~ 2 m	High
3.1 - 10 GHz	Ultra wide band	0 ~ 200 m	High

RFID technology can be used in different states of wireless automation systems. In an assembly line, the RFID tag attached to an automobile during production can be used to track its process. The products with RFID tags can be tracked throughout warehouses as well.

2.3.5. UWB

Ultra Wideband (UWB) technology provides a possibility to implement additional wireless communication in heavy narrow band oriented network environments. The reason is that UWB spreads over a wide frequency band so that the peak spectral power is not high enough to interference with narrowband radio. Over a short distance, to be precise less than 2 meters, the data rate of UWB can be achieved in a relatively high level, for example 500 Mbit/s.

According to Jantunen (2008), "The radio interface standardization is done within IEEE 802.15.3 and the full network solution with upper layers is standardized within WiMedia Alliance".

2.4. Concerns and Challenges

The implementation of wireless technology in industrial automation systems has remarkable benefits as discussed in previous sections. However, concerns and challenges also exist that have to be taken carefully into consideration.

2.4.1. Concerns

The reliability of wireless system is a critical concern. Radio transmission suffers from Bit Error Rates (BER) which is higher by order of magnitude compared to cables, where BERs range from 10^{-7} to 10^{-10} (Decotignie, 2002). If the BER is too high and there was not an error correction method applied, then there might be failure in the control loop of the system.

The vast majority of industrial plants are built within metallic structures which makes the radio environment challenging. The electromagnetic waves from wireless network have a possibility to cause current flows in those metallic structures. If the trigged current flow was large enough and was interrupted momentarily, sparks can occur resulting in ignitions in flammable atmosphere (Schultz, 2008).

2.4.2. Challenges

The major challenges in wireless automation are related to data and network security, interference, real-time delivery, power management, robustness, and effective utilization of limited bandwidth.

Sophisticated data encryption methods need to be introduced to keep the security of communication at a high level. If the ISM band was used as wireless channel, then the interference from other networks needs to be avoided. The wireless transmission latency should be controlled in very small range so that the control mechanism still works properly. Additionally, the robustness of the wireless automation system must reach a level which comes from the specifications and standards of that particular system.

3. WIRELESS SENSOR NETWORKS

3.1. Wireless Sensor Node

A Wireless Sensor Node is an electronic device with one or several sensors for monitoring various environmental and physical conditions at various places and times (**Figure 3** shows an example of wireless sensor node). Generally, a sensor node is composed of a Power Unit, Processing Unit, Sensing Unit, and Communication Unit. The Power Unit has the functions for supplying energy to the node. The Processing Unit is responsible for post processing of collected data from the sensing unit. The Sensing Unit is used for collecting data from surrounding environment by using one or several types of sensors, for example temperature, humidity, acceleration, light etc. The Communication Unit transfers and receives data in wireless manner (Vieira, Cunha & Silva 2006). Usually, the wireless sensor nodes are referred to as ‘nodes’.

Figure 3. An example of a wireless sensor node (Libelium World 2012).

A Wireless Sensor Network consists of many sensor nodes, and it can have many applications depending on the sensor type, for example fire detection in a forest, vibration detection in sheet metal manufacturing, humidity and carbon dioxide monitoring in advanced grain management, data logging from frequency converters in crane monitoring and many other.

3.2. IEEE 802.15.4 Standard

The vast majority of wireless sensor nodes use the IEEE 802.15.4 standard for wireless communication. The Local Area Network/Metropolitan Area Network (LAN/MAN) Standards Committee of the IEEE Computer Society developed the IEEE 802.15.4 Standard in 2003. It defines the Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low - Rate Wireless Personal Area Networks (LR - WPANs). There are three frequency bands in the physical layer of IEEE 802.15.4 standard: ISM band from 2400 to 2483.5 MHz, 868 MHz band and 915MHz band (see **Table 6**).

Table 6. Frequency band and data rate of IEEE 802.15.4 standard (LAN/MAN Standards Committee 2003).

PHY (MHz)	Frequency band (MHz)	Data rate (KB/s)
868	868 – 868.6	20
915	902 – 928	40
2450	2400 – 2483.5	250

The MAC sub layer of IEEE Standard 802.15.4-2003 handles all access to the physical radio channel. It is responsible for the following tasks: generating network beacons if the device is a coordinator, synchronizing to the beacons, providing a reliable link between two peer MAC entities, supporting device security and employing the Carrier Sense Multiple Access with Collision Avoidance (CSMA – CA) mechanism for channel access (IEEE 802.15.4 2003).

3.3. Operating Systems for Wireless Sensor Networks

There are numbers of different operating systems (OS) available for WSNs. The most frequently used ones include TinyOS (TinyOS 2014), FreeRTOS (FreeRTOS 2014), MANTIS Operating System (MOS 2014), and Contiki (Contiki OS 2014). The WSN developed in this thesis work is based on FreeRTOS Operating System.

TinyOS is an open source, Berkeley Software Distribution (BSD)-licensed operating system designed for low-power wireless devices, such as those used in sensor networks, ubiquitous computing, personal area networks, smart buildings, and smart meters (TinyOS 2014). TinyOS is considered as the most widely applied operating system among the commercially available WSN systems.

FreeRTOS is a class of Real Time Operating System (RTOS) also described as a real time kernel, or real time executive. It is designed to be small enough to run on a microcontroller, and it provides the core real time scheduling functionality, inter-task communication, timing and synchronization (FreeRTOS 2014).

The open source MOS was developed by research at the University of Colorado at Boulder. It is an embedded multi-threaded operating system for resource-constrained sensor nodes. It has features like developer friendly C Application Programming Interface (API), automatic preemptive time slicing for fast prototyping, diverse platform support, and energy-efficient scheduler for duty-cycle sleep mode and a small footprint (MOS 2014).

Contiki is an open source operating system which has particular focus on low-power wireless devices. It was created by Adam Dunkels in 2002 and has been further developed by world-wide teams from Atmel, Casco, Enea and so on. Contiki OS has the following features: multitasking kernel, pre-emptive multithreading, TCP/IP networking capability, a web browser and personal web server. Potential application includes street lighting systems, sound monitoring for smart cities, radiation monitoring systems and alarm systems (Contiki OS 2014).

3.4. Network Topologies of Wireless Sensor Networks

There are two different types of devices within a LR-WPAN network: Full Function Device (FFD) and Reduced Function Device (RFD). FFD is a relatively complex device that can operate as a Personal Area Network (PAN) coordinator. An RFD is intended for applications that are extremely simple, such as a light switch or a passive infrared sensor. Depending on the application requirements, the LR – WPAN may operate in either one of two topologies: Star Topology or Peer-to-Peer Topology as illustrated in **Figure 4** (IEEE 802.15.4 2003: 13).

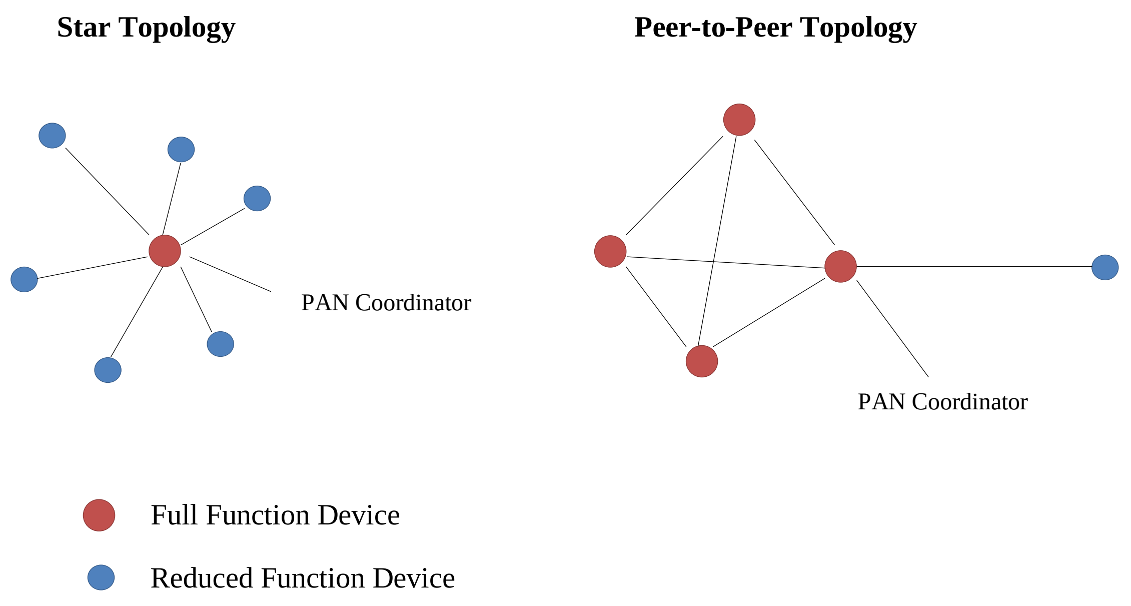


Figure 4. Network Topologies of IEEE 802.15.4 standard.

In Star Topology, the PAN coordinator which is a FFD communicates with numbers of RFDs. The PAN coordinator is the primary controller of the PAN, and it can be used as an initiator and terminator of the network. The RFD only communicates with its PAN coordinator and routing is not required. Some example applications that are based on

Star Topologies are home automation, personal computer peripherals, toys and games, and personal health care (IEEE 802.15.4 2003: 14).

In Peer-to-Peer (P2P) Topology, any two devices can communicate with each other inside the communication range. Thus, more complex topologies can be implemented such as mesh networking topology. Example applications using P2P topology are industrial control and monitoring, wireless sensor networks, asset and inventory tracking, intelligent agriculture and so on. It is also possible to implement ad hoc, self-organizing self-healing networks in P2P topology. In addition, functions like multi-hop routing can be added in network layer (IEEE 802.15.4 2003: 14).

3.5. Security

Security is always a critical aspect of all wireless network design including WSNs. The security threads exist in all levels of WSN systems, for example jamming might happen in physical layers of a network; an attack can cause collision in the data link layer; or even the unauthorized control packets in an application layer could mess up normal operation of WSN systems.

There are several methods introduced to achieve better security in WSNs, for example cryptography and authentication. Cryptography means encrypting data packets from third parties and decrypting it at the side of the receiver. The algorithm for encrypting and decrypting the message is called Key and there are in existence many ways of sharing the key (Pohjola 2008). According to Sastry & Wagner (2004), Network Shared Keying (NSK) model is the simplest way of sharing the key, in which all nodes in the network use one key. However, NSK mode is more vulnerable than other keying methods. Pairwise Keying (PK) model is more secure whereas each pair of nodes share a different key, but it needs more memory for better security performance. As a compromise between performance and resource requirement, the Group Keying (GK) model is a perfect solution. In GK model, a set of nodes in a group share a key, and normally the group is created based on the location of the nodes.

Authentication is applied to determine whether the received message is transmitted from a claimed node and it has not been tampered with during the transmission. Authentication is also used for accepting nodes to take part in the network. The messages only from authenticated nodes are accepted (Pohjola 2008).

In IEEE 802.15.4 standard based WSNs, there are four kinds of security services: Access Control, Data Encryption, Frame Integrity and Sequential Freshness. In Access Control service, the Access Control List (ACL) is implemented to name the authorized devices. Only the devices in the ACL list can interact with the network. Data Encryption service effectively protects data from eavesdropping since only the device with shared key can decrypt data. Frame Integrity is for preventing manipulation by invalid intrusions during transmission of data packets. A counter mechanism is introduced in Sequential Freshness service. The transmitter sets a counter value for every data packet in increasing manner. In the receiver side, any frames which have counter value equal or less than the previous counter value will be rejected (Li, Xue & Song 2010).

3.6. Quality of Service

WSNs need to fulfill certain Quality of Service (QoS) requirements in order to make the associated system work properly. As it is stated in Pohjola (2008), there are several QoS measures used in WSNs: Latency, delay jitter, packet loss, node lifetime and continuing operation.

Latency includes propagation delay, queuing delay, routing delay, etc. The maximum possible delay of WSNs should not exceed the application specified threshold. In traditional control theory, the delay is constant whereas in WSN system it varies. The variation of delay is difficult to handle, therefore WSNs should achieve a minimum amount of delay jitter.

Packet loss is a common issue in wireless networks. Ideally, WSNs are required to have zero packet loss, but in reality it is hardly possible and the packet loss need to be controlled within its lowest level. In the case of losing packets, additional functions like retransmission or estimation of missing packets are preferred.

Reliability is a vital aspect for any industrial systems. In order to achieve a reliable WSN for industrial applications, the life-time of nodes should be as long as possible. The ability of continuous operations of WSNs must be long enough, and self-recovery functionalities are needed to be able to automatically recover from unexpected failures in the network.

4. SYSTEM DESCRIPTION

This chapter describes the overall architecture of the developed data logging system.

4.1. Usage for Monitoring and Real Time Data Logging

The development of wireless data logging systems for frequency converters has two main purposes: Usage for Monitoring and Real-time Data Logging.

The wireless data logging system developed in this thesis work is able to collect data from several frequency converters. Thus, it can be adopted as a data collection part of a remote monitoring and reporting system in crane industries.

The wireless data logging system can be set to monitor the usage of the crane for several days. The safety of the crane system can be increased based on the analysis of changes in data. The emergency stops, overloads and other operator included safety issues can be captured and reported. This will further enable the user to address any problems and take corrective actions before any real incidents occur. Moreover, the collected data could also be used to optimize the maintenance costs.

This use case focuses on collecting data in a short period of time from several frequency converters mounted to same crane once the crane is operating. The collected data then can be used for analyzing the behavior of crane in that exact time duration. From these analyses, the user is able to see how the hoist is actually used and what is the estimated remaining service life of certain components of the crane. One can also utilize the collected data to plan and budget crane maintenance actions with greater confidence (KONECRANES 2012).

An example of a real-time data logging application is illustrated in **Figure 5**. In this example, three different speeds (hoist speed, trolley speed and bridge speed) were selected as parameters to be collected. Then at the user side, the collected data was analyzed and the hock location was calculated and plotted in a figure (see **Figure 6)**.

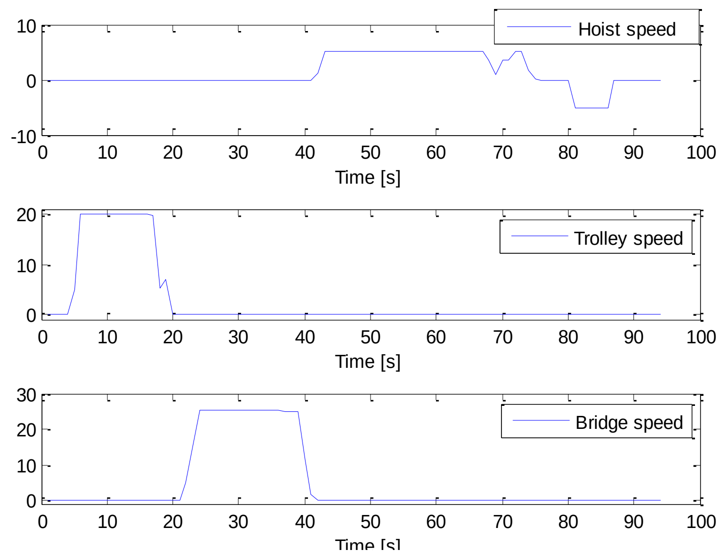


Figure 5. Collected speed data. (Kerovuori & Virrankoski 2013: 7)

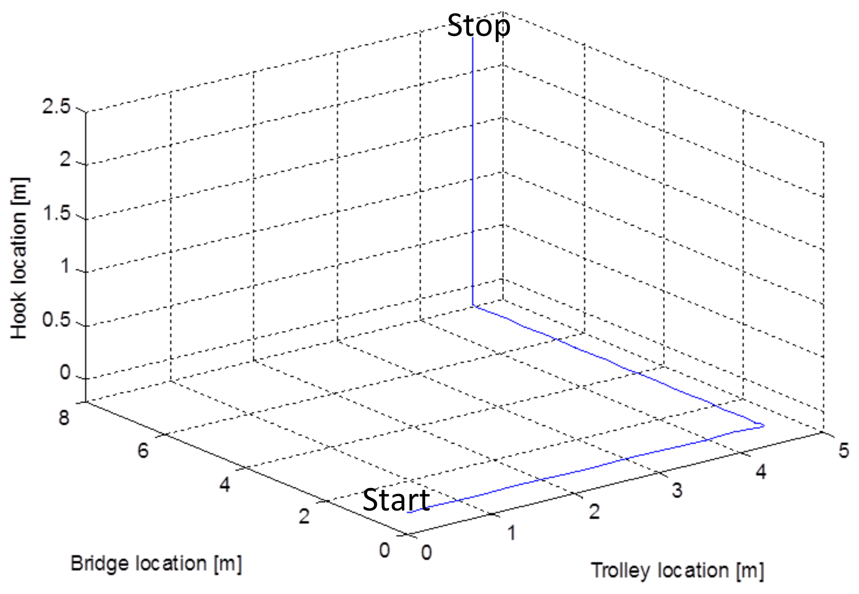


Figure 6. Calculated hock location. (Kerovuori & Virrankoski 2013: 8)

4.2. Hardware Components

The wireless data logging system discussed in this thesis work consists of three main hardware components: Frequency Converter, UWASA Node and Data Logging PC. The basic structure and unique features of each hardware component is explained in this section.

4.2.1. Frequency Converter

All frequency converters (see **Figure 7**) used in this thesis work are produced by Vacon which is an AC drive manufacturer headquartered in Vaasa, Finland. The frequency converter is used to control motors within a crane system. Several parameters that are located within a frequency converter's memory indicate the state of the motor. For example, the frequency value of the motor (memory index is 188) can be read from a frequency converter and further can be used for system analyzing purposes. A removable control panel (the black part in **Figure 7**) is connected to frequency converter through RS-232 serial port.

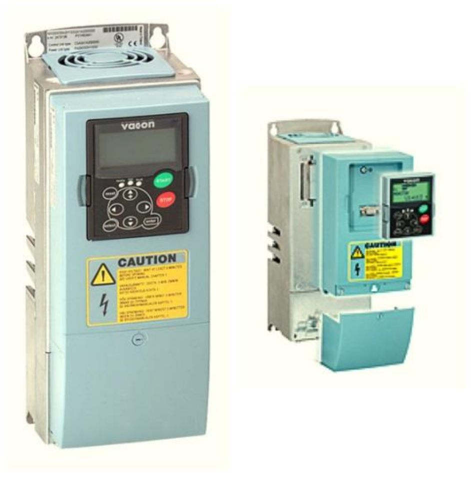


Figure 7. Vacon frequency converter.

4.2.2. UWASA Node

The UWASA Node is a modular and stackable wireless sensor platform, which could be used in various types of wireless automation applications. It is considered a generic platform because it provides adaptation and extension interfaces both for software and hardware to enable reuse of the same platform within different deployments. A third generation UWASA Node with development board is shown in **Figure 8** (Virrankoski 2012: 29).

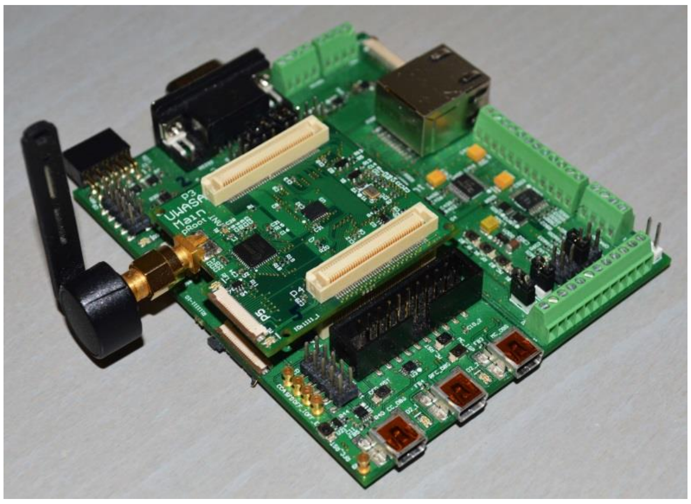


Figure 8. A third generation UWASA Node.

The modular and stackable hardware architecture is presented in **Figure 9**, and it is a significant feature of the UWASA Node. The Main Module consists of Main Controller (MC) and Radio Frequency Controller (RFC). MC provides a moderate level of computational power and memory, while RFC is responsible for wireless communication. Some simple slave modules can be added to the hardware stack according to the application demand. The signaling and the power supplies in the hardware stack are transferred through the Hardware Stack Connector (HSC).

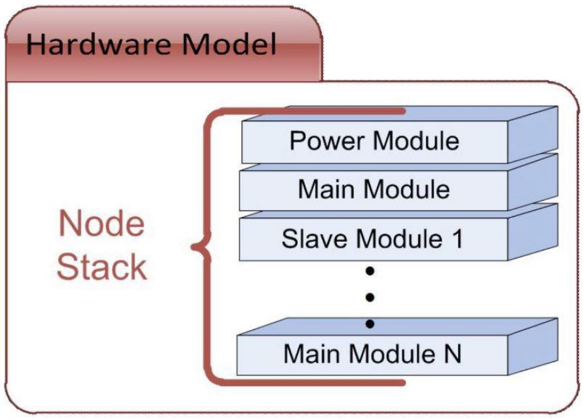


Figure 9. The hardware model of the UWASA Node (Virrankoski 2012: 32).

The UWASA Node software architecture is designed to have multiple levels so that the stackable hardware structure is fully abstracted. The **Figure 10** illustrates the Software Model of the UWASA Node.

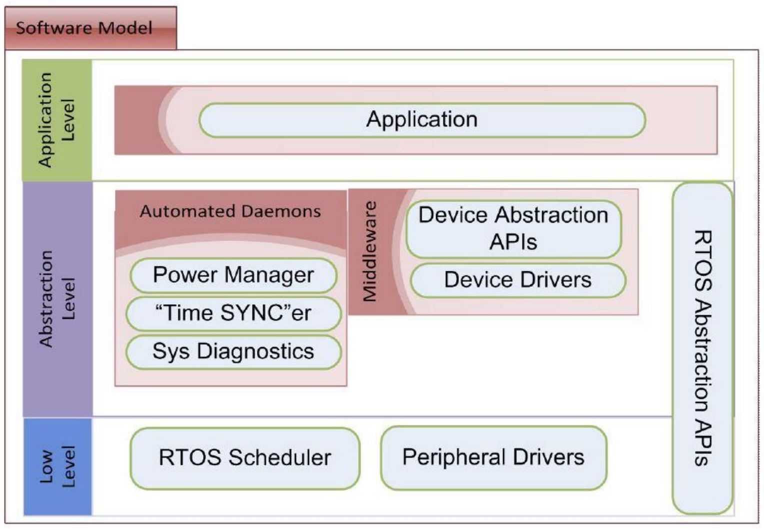


Figure 10. The Software model of the UWASA Node (Virrankoski 2012: 39).

The software architecture of the UWASA Node is divided into three layers: Low Level, Abstraction Level and Application Level. The Low Level software consists of the RTOS Scheduler and the Peripheral Drivers that are highly related to its hardware target. The automated daemons and middleware are the key components to reach abstraction from both RTOS and Peripherals, and they are created in the Abstraction Level. The last layer is the Application Layer that contains application software. The UWASA Node software can be reused in different deployments, so that the application layer software varies depending on the need of a specific application.

4.2.3. Data Logging PC

A Raspberry Pi embedded Linux computer (see **Figure 11**) was used as a Data Logging PC in this project. The Raspberry Pi is a credit-card-size single-board computer being developed in the United Kingdom since 2006. A typical Raspberry Pi computer comes with 700 MHz processing power and 512 MB of RAM. It also has multiple connections including Ethernet, USB, and HDMI. There are many Linux based open source operating systems available for development, for example Raspbian, Debian GNU, OpenELEC and so on.

Figure 11. Rasberry Pi computer model (Upton 2012).

4.3. Communication Links

After having all hardware components ready for deployment, a robust communication link must be established between each device. There are three types of communication links for sending a data packet all the way from a frequency converter to a data logging PC. This section specifies the transmission medium and the packet structure of each type of communication link.

4.3.1. Frequency Converter - Communication Node Link

The Communication Node refers to the UWASA Node that is connected to a frequency converter. The main task of this link is to read user defined parameter data from a frequency converter by using the UWASA Node. An RS-232 connection was applied as a physical medium for data transmission. A special protocol called Vacon Human Machine Interface (VHMI) was implemented in the UWASA Node to establish a solid communication with frequency converters. The VHMI protocol was designed by Vacon for interacting with their frequency converter products.

The Communication Node reads data from a frequency converter by sending Read Command. The data can be read either by Index or ID of the selected parameter. **Tables 7** and **8** represent Read by Index and Read by ID packet structures respectively. The Memory Location column in **Tables 7** and **8** indicates the parameter that selected by user. After receiving read request from the Communication Node, the frequency converter sends a response packet, which has a structure like **Table 9**. The Memory Content column in **Table 9** is the actual value of the user selected parameter.

Table7. Read by Index packet structure. (Kerovuori & Virrankoski 2013: 3)

Start Chars		**Command**	**Object**	**Read Command Identifier**	**Index**		**Number of Variables**		**Stop Chars**		**CRC**
DLE	STX	Read	Variable Object	Read by Index	Memory Location		In this application always 1		DLE	ETX	
0x10	0x02	0x40	0x0B	0x00	0x00 MSB	0xBC LSB	0x00 MSB	0x01 LSB	0x10	0x03	0x66

Table 8. Read by ID packet structure. (Kerovuori & Virrankoski 2013: 3)

Start Chars		**Command**	**Object**	**Read Command Identifier**	**ID**		**Number of Variables**		**Stop Chars**		**CRC**
DLE	STX	Read	Variable Object	Read by ID	Parameter ID				DLE	ETX	
0x10	0x02	0x40	0x0B	0x01	MSB	LSB	0x00 MSB	0x01 LSB	0x10	0x03	0x66

Table 9. Response packet structure. (Kerovuori & Virrankoski 2013: 4)

Start Chars		**Command**	**Object**	**Data Type**	**Data**		**Stop Chars**		**CRC**
DLE	STX	Response	Variable Object	Integer	Memory Contents		DLE	ETX	
0x10	0x02	0xC0	0x0B	0x03	0x00 MSB	0x00 LSB	0x10	0x03	0x52

4.3.2. Communication Node - Gateway Link

A simple star topology WSN was created by using these types of links to connect six Communication Nodes with a Gateway Node. A general packet structure was defined to regulate the packets for wireless communication in WSN as shown in **Table 10**.

Table 10. General packet structure for wireless communication.

Destination Node ID	Command Specification	Data	CRC
2 bytes	1 byte	N byte	1 byte

The Command Specification column in **Table 10** represents the type of wireless packet. There are six different types of packets in wireless communication links which are shown in **Table 11**.

Table 11. Command types in Communication Node - Gateway link.

Command Types	CS
Node Check	1
Configuration	2
Start	3
Stop	4
Read	5
Response (* from communication node to GW)	6
Response Request (* from GW to communication node)	6

The MAC address of the Gateway Node was set to 0x0000 and the addresses of the Communication Nodes can be selected freely. In the developed network architecture, the addresses of six nodes were set from 0x0001 to 0x0006. As for the broadcast packets, the destination node ID was set to 0xFFFF.

There are four different types of commands used in communication in this WSN: Node Check Command, Configuration Command, Read Command and Response Request Command. Node Check Command is considered as an additional function of this data logging system by which the user is able to check if one specific node is alive in the network. The Configuration Command is set for configuring the Communication Nodes including Read Command Type, Memory Index or ID and Number of Variables. Acknowledgement functionality was implemented in both types of commands to insure the desired packet is received correctly at the Communication Node. **Table 12** represents the packet structure of Configuration Command. The packet structure of Node Check and Configuration Command can be found in Appendix 1.

Table 12. The Configuration Command in Communication Node - Gateway Link.

Destination	**CS**	**Data**							**CRC**
0x0001	0x02	RCI1 (1 byte)	MI1 (2byte)	NV1 (1byte)	RCI2 (1 byte)	MI2 (2 byte)	NV2 (1byte)	...	1 byte

Once the data logging system starts working, the Gateway Node will continuously send Read Command to all Communication Nodes as a broadcast message. The reception of the Read Command is the moment that a Communication Node starts to collect data from frequency converters. After receiving data from frequency converters, the Communication Node will not immediately transmit the data to the Gateway Node. Instead all Communication Nodes will wait for another command for transmission permission which is called Response Request Command (RRC). The main reason for using this mechanism is to avoid collision in WSN. The RRC is transmitted in a sequence from node No.1 to node No.6. The packet structure of Read Command and Response Request Command can be found in Appendix 1.

4.3.3. Gateway - Data Logging PC Link

This communication link is established between Gateway Node and Data Logging PC over USB connection. There are two purposes for this communication. The first task covers the transmission of collected data from WSN to Data Logging PC as well as acknowledgement signals. The second task is the receiving of various types of commands from a User Interface (UI) device through Data Logging PC.

There are five different types of packets used in this application: Node Check Command, Configuration Command, Start Command, Stop Command and Response Packet. All four types of packets are following a general packet structure that is specified for this communication application, as shown in **Table 13**.

Table 13. The general packet structure in Gateway - Data Logging PC link.

Start	**Length**	**Node ID**	**CS**	**Data**	**End or CRC**
0xAA	1 byte	2 bytes	1 byte	n bytes	0x03

As it has been introduced in previous section, the Node Check Command is responsible for checking node states in WSN. Also, the Configuration Command has the same function in setting nodes' parameters. However, the only difference is that the logging interval is defined in this Configuration Command (see **Table 14**).

Table 14. The packet structure of configuration command in Gateway - Data Logging PC link.

Start	**Length**	**Node ID**	**CS**	**Data**							**End or CRC**
				Logging Interval	Read Command Identifier 1	...	Read Command Identifier 10	Memory Index 1	...	Memory Index 10	
0xAA	1 byte	2 bytes	0x02	2 bytes	1 bytes		1 bytes	2 bytes		2 bytes	0x03

The start command and stop command are reserved for initiating and terminating the whole system respectively. Once a start command is received from the data logging PC, the Gateway Node will continuously transmit read command as a broadcast message to all Communication Nodes. It was stated in section 4.1 that this wireless data logging system has two different working modes: continuous logging mode and log only changes mode. The logging mode setting is placed in the 'Data' column of start command as it is shown in **Table 15**. The 'Data' byte has to be set to 0x01 for continuous logging and 0x02 for 'log only changes'. The detailed packet structures of node check command, stop command and response packet can be found in Appendix 2.

Table 15. The start command in Gateway - Data Logging PC link.

Start	**Length**	**Node ID**	**CS**	**Data**	**End or CRC**
				Start command	
0xAA	1 byte	0xFFFF Broadcast ID	0x03	0x01 continuous logging 0x02 log only changes	0x03

5. SOFTWARE ARCHITECTURE

All of the software related topics are addressed in this chapter.

5.1. Software Architecture of the UWASA Node

Generally, nodes and actuators in WSN are expected to transmit wireless signals at the same time when they execute computation or control tasks. To be able to do that, a suitable OS is needed in the sensor node. There are many types of OSs available on the market for embedded devices. However, the task management structure of most of the OS introduces a certain amount of delay. The creation of the UWASA Node was targeted to wireless control applications, so that the delay could be avoided. As a result, the FreeRTOS was chosen as the operating system of the UWASA Node. The FreeRTOS operating system is based on preemptive multitasking so that there is no delay caused by task management. FreeRTOS API provides the basic operating system functionalities, and the detailed information about FreeRTOS can be found in its website (FreeRTOS 2014).

The software model of the UWASA Node consists of three levels: Low Level, Abstraction Level and Application Level (see **Figure 10)**. The hardware related peripheral initializations and configurations are performed in Low Level, for example, Universal Synchronous and Asynchronous Receiver/Transmitter (UART) port configuration and data structure initialization, and Interrupt Service Register (ISR) function implementations. The Abstraction Level consists of power management, system diagnostics, time synchronizer, device abstraction APIs, and device drivers. The power management mechanism provides four types of power modes: normal operation mode, idle mode, sleep mode and power-down mode. System diagnostics are designed to check the functionality of stack components. The abstraction API is hardware independent and it provides communication with device drivers. The highest level of software, known as Application Level, consists of application specific tasks.

5.1.1. Buffer APIs

The buffer structure was implemented in the UWASA Node software for data storage and processing. The software architecture of the UWASA Node consists of two types of buffer API: FIFO Buffer API and Bufbuffer API.

The FIFO (First In First Out) Buffer API is defined as a structure, and it contains data array, a head pointer and a tail pointer. Data array indicates the location of the buffer memory. Head pointer points to the memory location where the upcoming data will be stored. Tail pointer points to the memory location of the first data. **Figure 12** illustrates the structure of FIFO Buffer API.

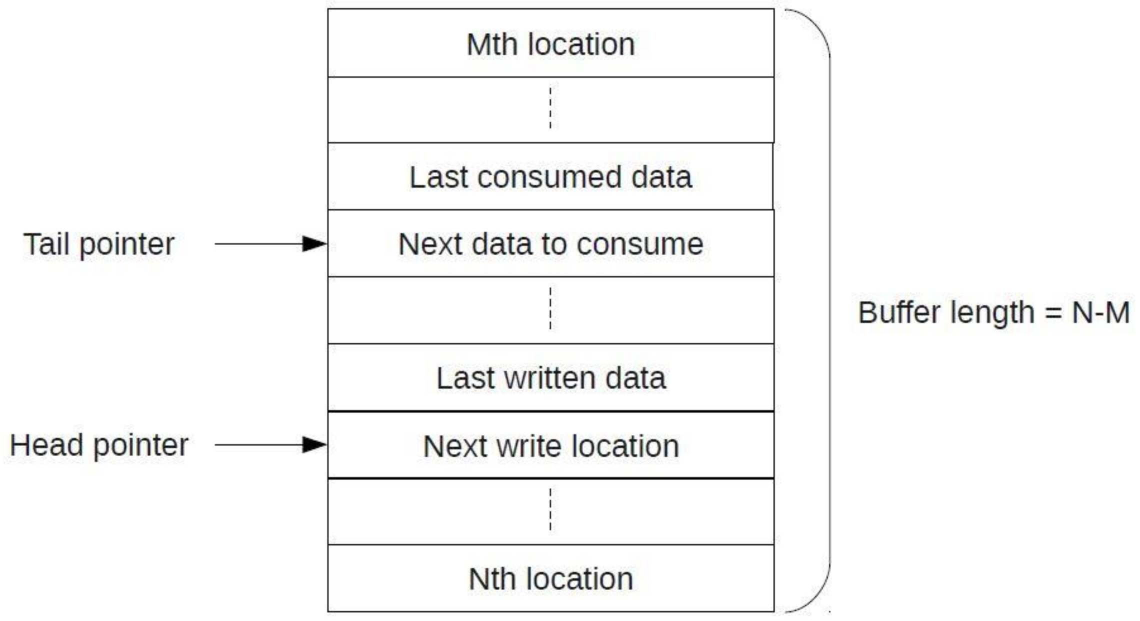
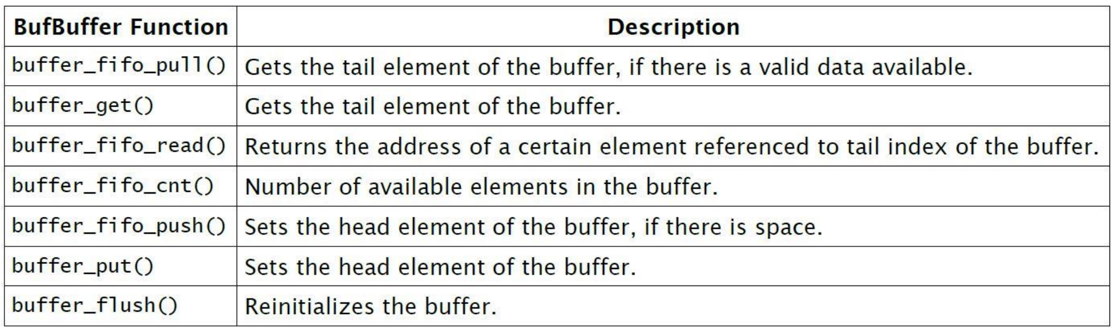


Figure 12. FIFO Buffer structure (Yiğitler 2012: 37).

There are seven functions implemented in FIFO Buffer API as shown in **Table 16**. In addition, a binary semaphore mechanism is used as a guard for accessing the resource between producer task and consumer task. It works in such a way that the consumer task calls buffer_wait_ready() function to check if the buffer is ready, if yes, it takes the semaphore and continues its operation, if not, it gets blocked.

Table 16. FIFO Buffer API functions (Yiğitler 2012: 39).



BufBuffer Function	Description
buffer_fifo_pull()	Gets the tail element of the buffer, if there is a valid data available.
buffer_get()	Gets the tail element of the buffer.
buffer_fifo_read()	Returns the address of a certain element referenced to tail index of the buffer.
buffer_fifo_cnt()	Number of available elements in the buffer.
buffer_fifo_push()	Sets the head element of the buffer, if there is space.
buffer_put()	Sets the head element of the buffer.
buffer_flush()	Reinitializes the buffer.

Another type of buffer structure is called Bufbuffer API. It is constructed by circular buffer of FIFO Buffers as shown in **Figure 13.** The data storage and processing is more efficient with Bufbuffer API as the producer task can fill a buffer independent of the consumer task. According to Virrankoski (2012), the Bufbuffer API signals the consumer about the availability of ready buffer, while producer continues to fill the next empty buffer in the circular queue. The consumer task reads or manipulates data from the ready FIFO buffers.

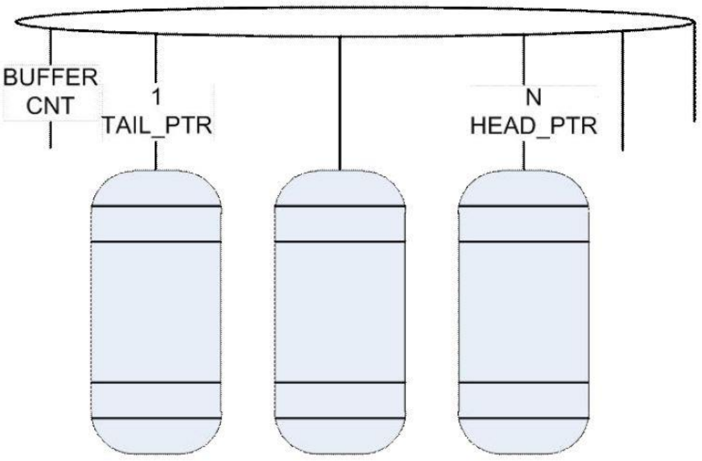
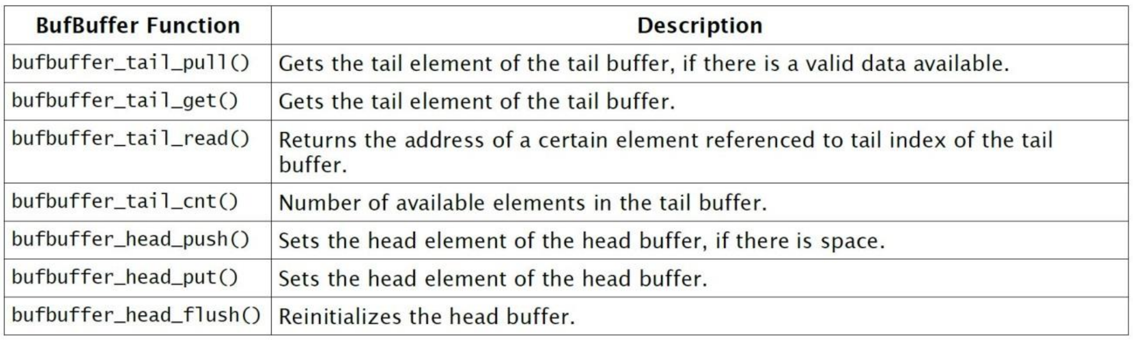


Figure 13. Bufbuffer data structure (Virrankoski 2012: 42).

Bufbuffer API also has seven functions as listed in **Table 17**. The producer task fills the buffer and then set the current FIFO buffer in ready state by calling bufbuffer_goto_next() function. After that, it moves to the next buffer and does the same operation. The consumer task, however, calls bufbuffer_wait_tail_ready() function to block itself until a buffer is set ready. After the buffer is ready, the consumer task calls bufbuffer_get_ready() function to get the oldest ready buffer by increasing the tail pointer.

Table 17. Bufbuffer API functions (Yiğitler 2012: 41).



BufBuffer Function	Description
bufbuffer_tail_pull()	Gets the tail element of the tail buffer, if there is a valid data available.
bufbuffer_tail_get()	Gets the tail element of the tail buffer.
bufbuffer_tail_read()	Returns the address of a certain element referenced to tail index of the tail buffer.
bufbuffer_tail_cnt()	Number of available elements in the tail buffer.
bufbuffer_head_push()	Sets the head element of the head buffer, if there is space.
bufbuffer_head_put()	Sets the head element of the head buffer.
bufbuffer_head_flush()	Reinitializes the head buffer.

5.1.2. Event APIs

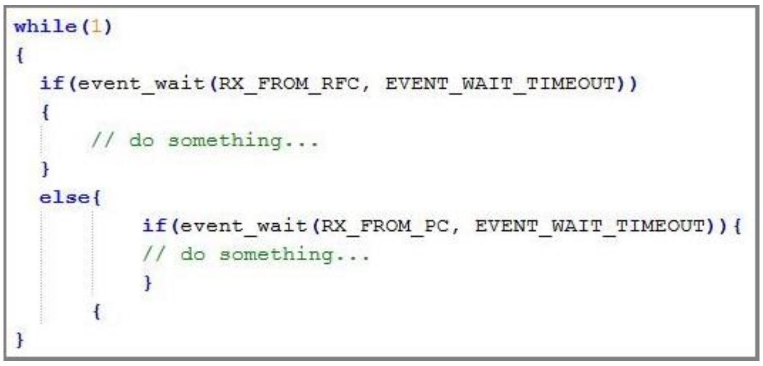
The hardware component of the UWASA Node includes some external devices, for example UART, USB, Timer and so on. From the Microcontroller point of view, the beginning point of the functionality of the external devices is to inform the processor about an occurred event. Microcontroller then provides suitable interrupt mechanisms to handle the situation. It is also very necessary to inform the related application tasks about the occurrence of an event. The Event APIs are designed for these purposes.

The Event APIs of the UWASA Node consists of three parts: Events, Event Groups and Event Manager. An Event has the following fields as shown in **Table 18**. The eventId is the identifier of the event and it is usually an integer number between 0 and 10. The eventGroupId is the identifier of the group which the event belongs to. The vParam field does not have any meaning before the event is defined. pParam is a void pointer to the parameter of the occurred event. The callback field is the callback function, and it can be registered at any moment during runtime. The signal is the semaphore which can block the tasks waiting for this event to occur.

Table 18. Event fields. (Yiğitler 2012: 42)

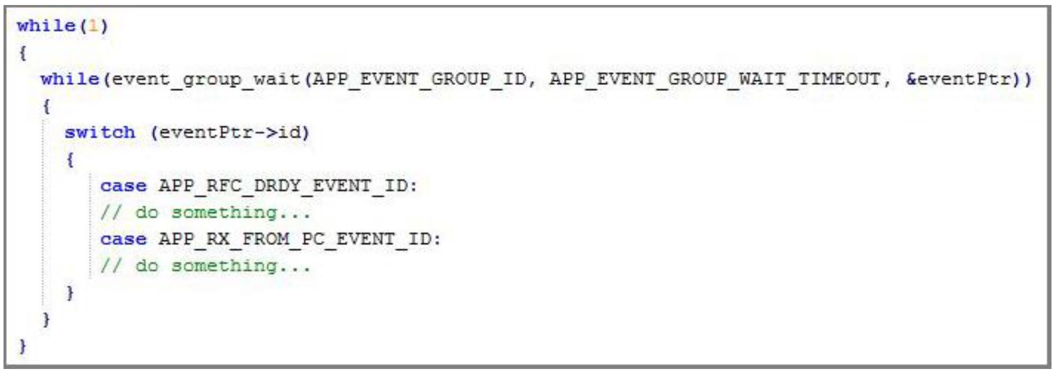
Event Fields	
id	eventId
groupId	eventGroupId
vParam	UINT
pParam	dptr_native_t
callback	eventCallback_t
signal	bSemaphore_t

There might be multiple events from the same peripheral. For the convenience, multiple events can be handled in the way of group. In the UWASA Node's software design, the event groups is implemented to group different events. For example, when considering the software design of the Gateway Node in this thesis work, there are two events: 'receiving data from RFC' and 'receiving data from data logging PC'. There are two different ways to handle these events: create two events separately or create them in one event group (see **Figures 14 and 15**). The event manager is used to initialize the events and event groups. This initialization is usually done at the beginning of the application by calling event_manager_init() function.



```
while(1)
{
  if(event_wait(RX_FROM_RFC, EVENT_WAIT_TIMEOUT))
  {
      // do something...
  }
  else{
            if(event_wait(RX_FROM_PC, EVENT_WAIT_TIMEOUT)){
            // do something...
            }
      {
}
```

Figure 14. Example of creating two events separately.



```
while(1)
{
  while(event_group_wait(APP_EVENT_GROUP_ID, APP_EVENT_GROUP_WAIT_TIMEOUT, &eventPtr))
  {
    switch (eventPtr->id)
    {
       case APP_RFC_DRDY_EVENT_ID:
       // do something...
       case APP_RX_FROM_PC_EVENT_ID:
       // do something...
    }
  }
}
```

Figure 15. Example of creating two events in one event group.

5.2. Software Design of the Communication Node

The WSN used in the data logger application consists of six Communication Nodes and one Gateway Node. All the wireless sensor nodes are UWASA Nodes. However, the downloaded softwares on the Communication Nodes and Gateway Node are different based on their specific tasks. Sections 5.2 and 5.3 describe the software architecture of the Communication Nodes and the Gateway Node respectively.

The main task of each Communication Node is to read selected parameter data from the frequency converter and transmit the data to the Gateway Node. Additional functionalities are added for the need of configuring the node settings and checking the node states. The flowchart in **Figure 16** illustrates the overall software architecture of the Communication Node. There is only one task in the Communication Node software, and the main() function is considered as the starting point of the software. The initialization of the target board and RFC, creation of task are declared in main() function. The scheduler is the last part of the main() function, and it starts the operation of the task.

The task in the Communication Node software is called vVHMIMaster. The RFC configuration and VHMI communication speed setting is done at the beginning of the task. In order to achieve the minimum reading time from the frequency converter, the communication speed between the node and the frequency converter is set to the highest value (vhmi_115200). At this point, the Communication Node is in Wait Command state, and it is ready to receive data from the Gateway Node (see **Figure 16**). There is a possibility to receive one of the four types of command sent from the Gateway Node: node check, configuration, read, and response request. All the operations triggered by each command follow the flowchart illustrated in **Figure 16**.

One special design that needs to be highlighted here is the response request command. The Communication Node receives data from the frequency converter and operates data processing functions, but it will not send the data to the Gateway Node until it receives the response request command. The idea behind this mechanism is to transmit data in time sequence, so that the potential collision is avoided in WSN. In the Communication Node software flowchart (**Figure 16**), FC stands for Frequency Converter and GW stands for Gateway.

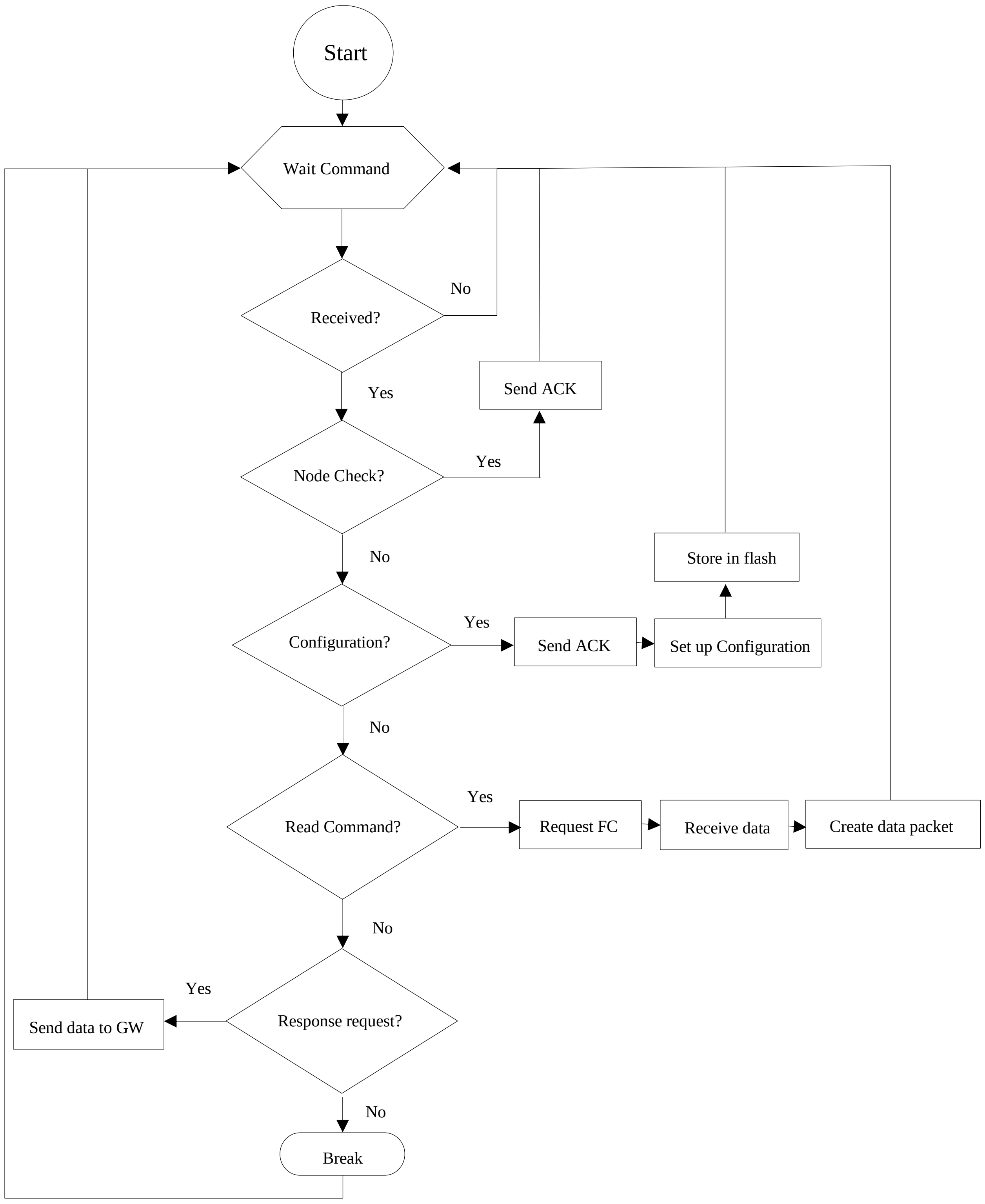


Figure 16. The software flowchart of the Communication Node.

5.3. Software Design of the Gateway Node

The software architecture of the Gateway Node is more complex than the architecture of the Communication Node. It is not simply because the Gateway Node interacts with WSN and Data Logging PC but also because it performs more data processing. In addition to that, the packet structure in wireless communication and the packet structure in the wired communication between the Gateway Node and the data logging PC are different. For this reason, the received data needs to be re-packeted in the Gateway Node before sending it to the data logging PC.

The software design of the Gateway Node abstracts two main goals. One is to receive different types of commands from the data logging PC and transmit them to the Communication Nodes. The second goal is to receive collected data from the Communication Nodes and transmit them to the data logging PC. A software flowchart in **Figure 17** describes the overall software architecture of the Gateway Node. The structure of the main() function is the same as in the Communication Node except there are three semaphore implementations added.

There are two task implementations in this software architecture: pcgwMaster and pcgwSend. After configuration of RFC, the Gateway Node enters the wait event state, in which it receives data either from WSN or the Data Logging PC. If the data was received from the WSN, the Gateway Node will process the data and send it further to the Data Logging PC. For the commands that are received from the data logging PC, the Gateway Node will proceed with corresponding operations according to the flowchart presented in **Figure 17**.

There is a 100 millisecond time interval between read command and response request command (see **Figure 17**). The main purpose of this design is to give enough time for the communication between frequency converter and Communication Node. It has been discovered during the development that the average time it takes to read 10 parameters of data from the frequency converter varies from 50 to 90 milliseconds.

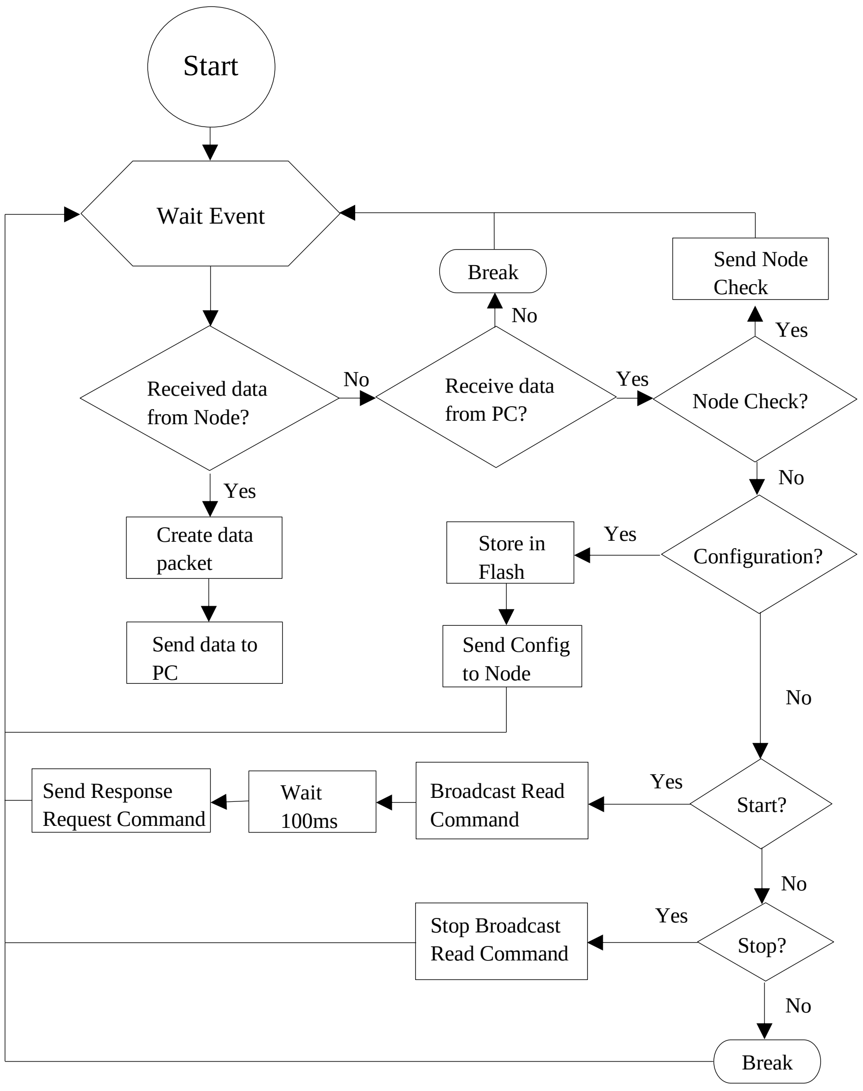


Figure 17. The software flowchart of the Gateway Node.

5.4. Software Design of the Data Logging PC

The hardware device used as the data logging PC was a Raspberry Pi with embedded Linux. The data logging PC plays a bridge role between WSN and UI (see **Figure 1**). A database was used to save certain collected data; therefore the data logging PC also needs to interact with the database.

The software abstraction is illustrated in **Figure 18.** It contains four components: control module, WSN interaction module, UI interaction module and database interaction module. The main goal of the control module is to ensure coordination between the various software components by performing task controlling and so on. The WSN interaction module and UI interaction module are responsible for data tranceiving between WSN and UI respectively. The database interaction module is designed for the purpose of saving some collected data to database as well as providing data from database if needed.

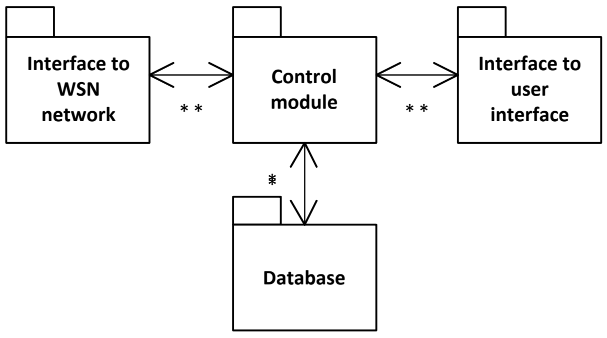


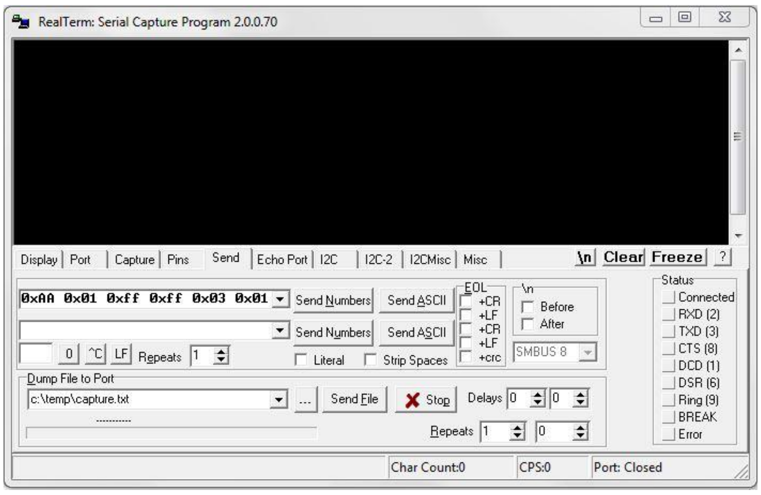
Figure 18. The software architecture of the data logging PC (Kerovuori & Virrankoski 2013: 7).

6. EXPERIMENTS AND RESULTS

This chapter describes the experimental set ups of the developed wireless data logging system. The corresponding test results as well as their analysis are provided. Several issues have been discovered during the experiments; therefore possible system improvements are also introduced.

6.1. Graphical Test User Interface

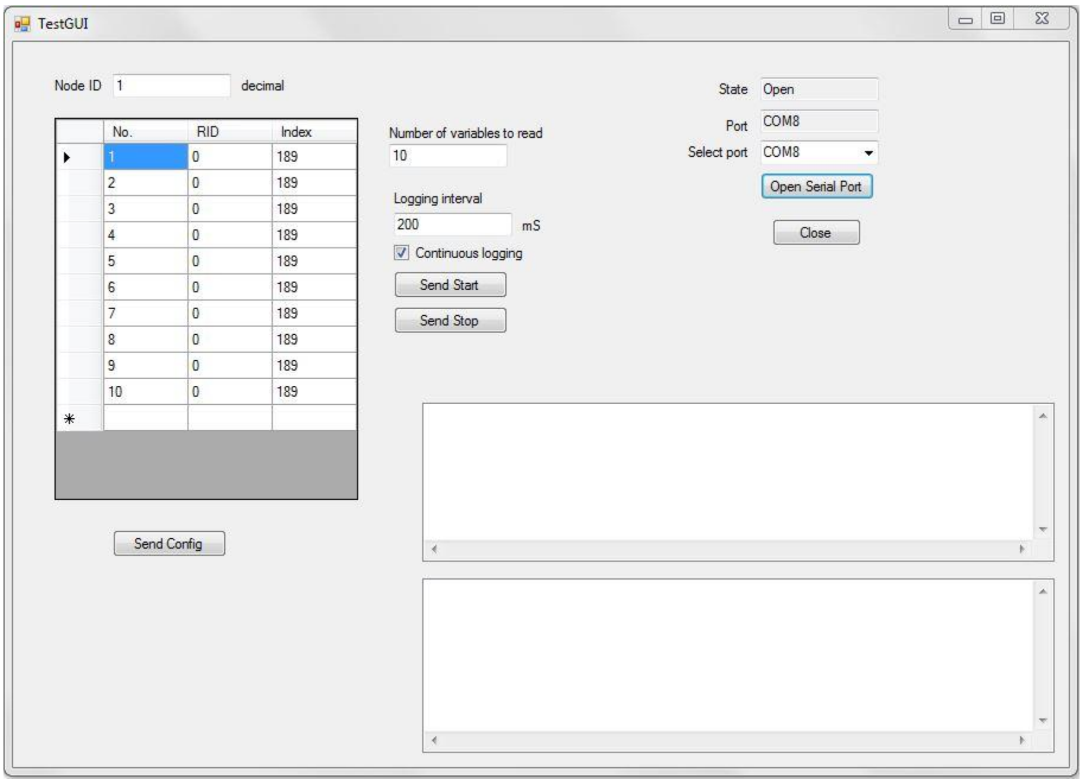
In the case of each experiment, the configuration of the Communication Node and the Gateway Node has to be done before starting the data logging system. The settings include ‘number of parameters to be logged’, ‘parameter indexes’, ‘read by index or ID’, ‘logging interval’ and ‘logging mode’ (continuous logging or log only changes). All these configurations can be done by using the RealTerm Serial Capture Program as shown in **Figure 19**. However, it is not convenient to type all the bytes of configuration fields for each test. For this reason, a simple graphical test user interface (GTUI) was implemented.



```
0xAA 0x01 0xff 0xff 0x03 0x01
c:\temp\capture.txt
Char Count:0    CPS:0    Port: Closed
```

Figure 19. RealTerm: Serial Capture Program for starting, stopping and configuring the wireless data logging system.

Figure 20 presents the GTUI for this wireless data logging system. The serial port selection, port name and port state are located at the right hand side of GTUI. After the serial port is connected, a user can select which one of the nodes would be configured by putting a decimal number at Node ID box. The Node ID is from 1 to 6 since there are only six Communication Nodes in this system. Then one must set a value for the 'Number of variables to read' field, and user can select up to 10 parameters to be read. Next one must define the logging interval for whole system. The minimum logging interval is 200 milliseconds. User can also select 'log only changes' mode by unclicking the arrow in 'Continuous logging' field. By default the data logging system is set to work at 'continuous logging' mode. After all these configurations are done, the user can start the whole system by clicking 'Send Start' button. For stopping the system, one can simply click 'Send Stop' button. The white blank text field in the bottom right of the **Figure 20** is designed for the displaying of the received data.



Node ID 1 decimal

No.	RID	Index
1	0	189
2	0	189
3	0	189
4	0	189
5	0	189
6	0	189
7	0	189
8	0	189
9	0	189
10	0	189

Number of variables to read 10

Logging interval 200 mS

[✓] Continuous logging

State Open

Port COM8

Select port COM8

Figure 20. The graphical test user interface for wireless data logging system.

6.2. Experiments at Riihimäki Testsite

This wireless data logging system was tested twice at the research center of KONECRANES in Riihimäki. Data was collected in a wireless manner from six Vacon frequency converters which were installed into one of KONECRANES's lifting products. Six programmed Communication Nodes were used to collect data from frequency converters. The Communication Nodes were connected to frequency converters over RS-232 cable as it is shown in **Figure 21**, and they transmitted the data wirelessly based on the transmission requests they received. USB cable connection was used to power up the nodes directly from frequency converters. As a consequence, there was no need for any other power source in the UWASA Nodes.

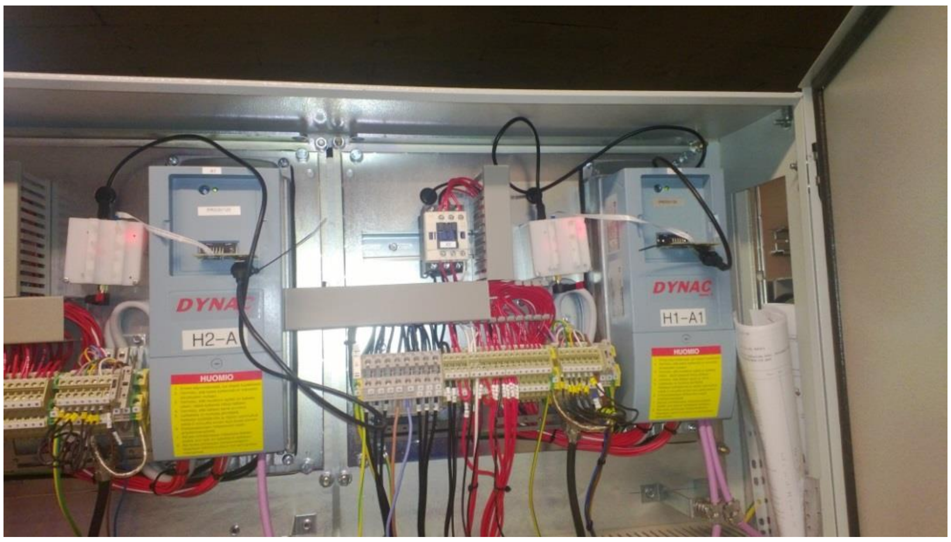


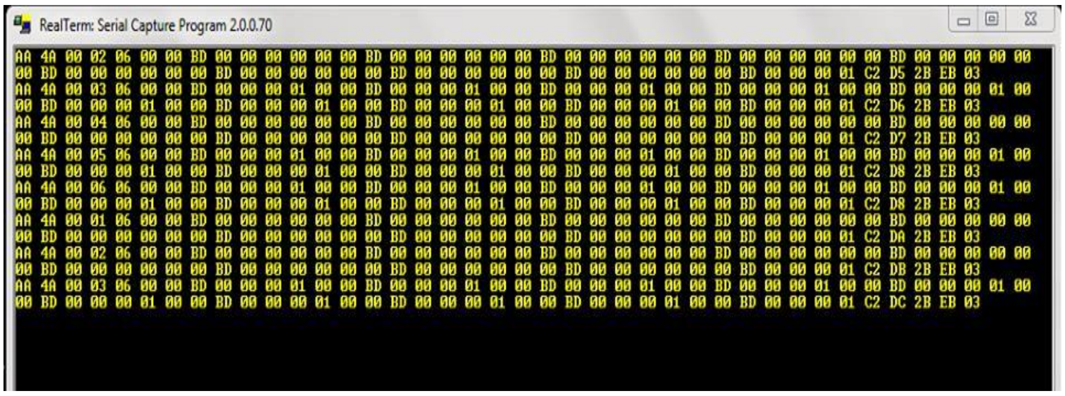
Figure 21. Test setup at Riihimäki. Big blue box device is frequency converter and the UWASA Nodes which are used as Communication Nodes are sealed to white boxes.

The Gateway Node was connected to embedded Linux computer (Raspberry Pi) for starting, stopping and configuring the whole system. The physical location of the Gateway Node was fixed, but the Communication Nodes were moving with the crane.

The wireless transmission range between the Gateway Node and Communication Nodes varied roughly from 3 to 30 meters.

The first experiment in Riihimäki failed because during the experiment it was discovered that the Communication Node was not able to read data from frequency converters. Later, investigations were held to fix the problem in this specific communication link (Communication Node-Frequency Converter Link). Eventually the issue was found, and the problem was located in soldering the RS-232 cable. This problem was taken care of and the second test was successful.

In the second test, the wireless data logging system was able to collect the values of the selected 10 parameters from each of the 6 frequency converters once in 200 milliseconds. The system worked smoothly even though the Communication Nodes were moving with the crane. The total throughput of the system at the Gateway Node was 16.8Kb/s. **Figure 22** presents the example of collected data from one of the experiments in RealTerm Serial Capture Program.



```
AA 4A 00 02 06 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00
00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 01 C2 D5 2B EB 03
AA 4A 00 03 06 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00
00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 C2 D6 2B EB 03
AA 4A 00 04 06 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00
00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 01 C2 D7 2B EB 03
AA 4A 00 05 06 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00
00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 C2 D8 2B EB 03
AA 4A 00 06 06 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00
00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 C2 D8 2B EB 03
AA 4A 00 01 06 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00
00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 01 C2 DA 2B EB 03
AA 4A 00 02 06 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00
00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 00 00 00 BD 00 00 00 01 C2 DB 2B EB 03
AA 4A 00 03 06 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00
00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 00 00 BD 00 00 00 01 C2 DC 2B EB 03
```

Figure 22. An example of collected data displayed in RealTerm Serial Capture Program.

The experiment results indicate that each communication link as well as the software implementations were working well. However, there were two main challenges in the system performance: packet loss and system reliability.

6.3. Packet Loss Analysis

In this section, the packet loss related issues in wireless transmission are discussed based on the experimental results. At the first set of experiments, the packet loss was significant, approximately 10%; therefore a resend mechanism (See **Figure 24**) was implemented to reduce the packet loss percentage to an acceptable range around 1%.

In an ideal case, the Gateway Node should receive one Response Packet from a Communication Node for each Response Request Packet. However, there is always a certain level of packet loss in wireless transmission. The main reasons for losing packets can be categorized into two main categories: hardware characteristics and wireless channel properties. It has been discovered that in some of the UWASA Nodes the connection between the hardware board and antenna was broken. This could significantly reduce the sensitivity of antenna especially for longer distance wireless transmission. This problem was fixed by soldering the critical connection better between the antenna and the board. The wireless channel is changing over the time and it is not possible to adjust it as desired. The factory environment can be especially challenging radio environment, since it can contain, for example, sharp edged metals, and interference caused by other wireless communication systems.

The packet loss of wireless transmission was calculated as:

$$packet\ loss\ percentage = \frac{number\ of\ lost\ packet}{total\ number\ of\ Response\ Request} \times 100\%$$

The '*number of lost packet*' and '*total number of Response Request*' were generated in the packet which was sent from Gateway Node to Data Logging PC. Six bytes before the end byte (0x03) was reserved for this purpose, two bytes for '*number of lost packet*' and four bytes for '*number of Response Request*'.

All the performance measurements in terms of packet loss were done at University of Vaasa, after the Riihimäki experiments. Due to the limitation of testing environment, the applied distance between Gateway and each Communication Node is constant, which has a shortest value of 0.5 miters and a longest value of 3 meters. **Figure 23** shows the results of packet loss measurements from 30 individual tests, which was done before making any system improvements. There are 10 packet loss measurements in each individual test, and the average number of these 10 measurements was taken as the packet loss percentage of this individual test. The red line in **Figure 23** indicates the average packet loss of these 30 individual tests. As it can be seen, the overall average packet loss in these individual tests was 9.92%, and the overall standard deviation was 0.485. This packet loss was significant (almost 10%).

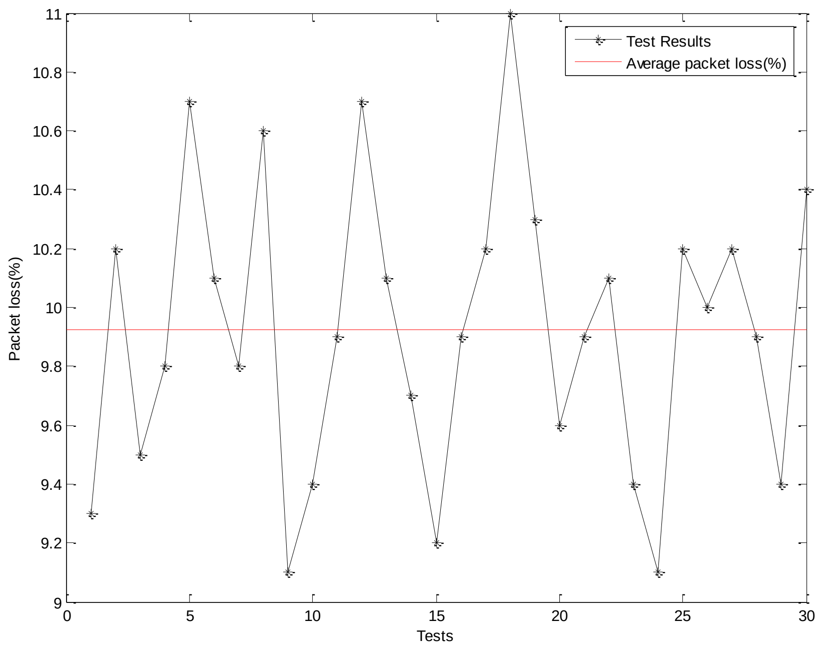


Figure 23. The packet loss measurement results before system improvements.

In order to overcome the packet loss issue, a resend mechanism was designed and implemented. **Figure 24** describes the idea of the implemented resend mechanism.

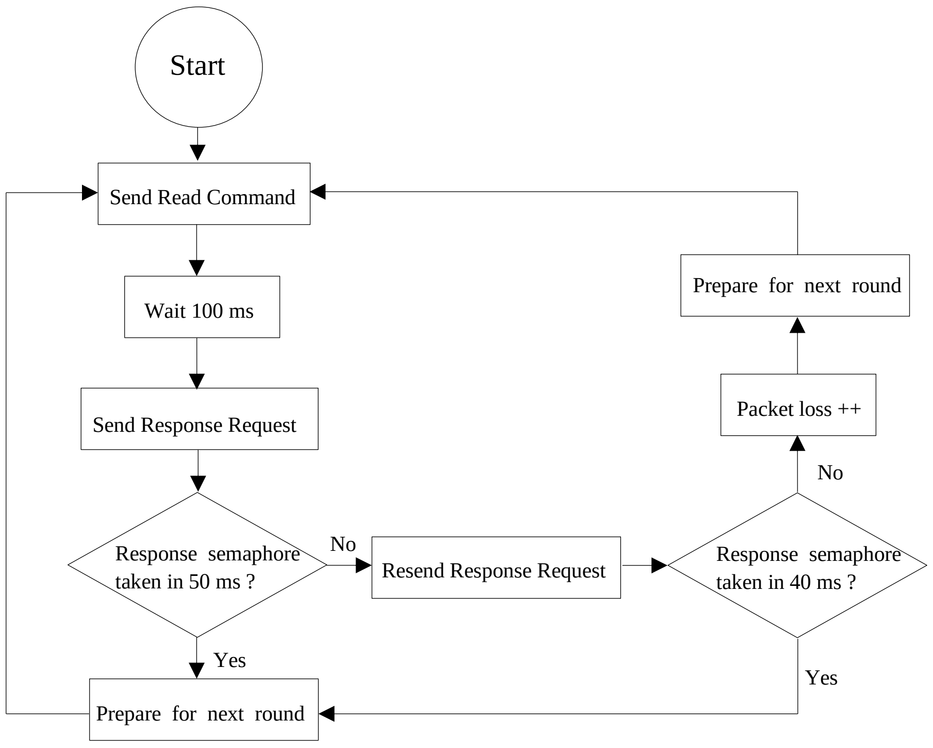


Figure 24. Software flowchart of '*resend'* function.

A semaphore called response semaphore was created in '*resend*'. It is given after the Gateway Node receives a response packet from the Communication Node. The Gateway Node broadcasts read command to all communication nodes, and then after a 100 millisecond time interval, it sends response request command to each node starting from Node 1. If the Gateway Node can take the response semaphore in 50 milliseconds, which means it has received the response packet from Node 1, then it will send response request command to Node 2. If the Gateway Node can not take that Response Semaphore in 50 milliseconds then it will send another response request command to that specific Communication Node and wait for 40 milliseconds to take the response semaphore. If the response semaphore was taken at the second time, it will send response request command to the next Communication Node, otherwise it is considered as packet loss, and the packet loss number will be increased.

The further experiments indicated that the implementation of the resend mechanism significantly reduced the packet loss. **Figure 25** illustrates the results of packet loss measurement from 30 individual tests after the system improvements. Test results shows that the resend mechanism reduced the overall average packet loss to 1.22% which is remarkably lower than the previous test results. Also, the overall standard deviation of packet loss is 0.418, which is also slightly lower than previous outcome.

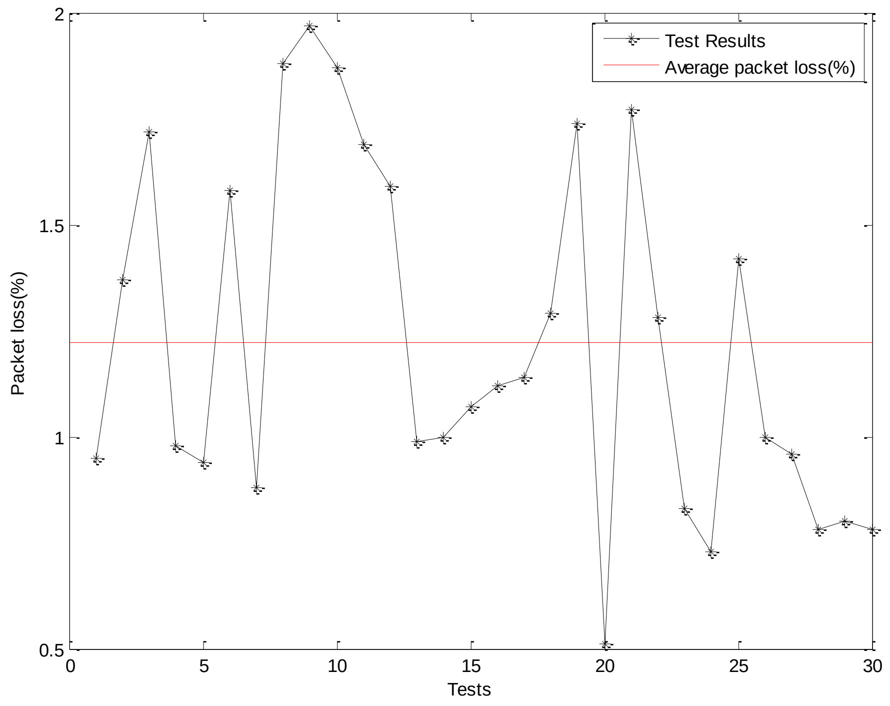


Figure 25. Average packet loss after implementing the resend mechanism.

6.4. System Reliability

Reliability was another challenge in this thesis work. It was discovered in the experiments that the system operation was not fully reliable. There was a random node that was suffering from software upsets at random time moments. **Figure 26** illustrates the observed reliability of the system. This statistic was taken from 30 individual tests, and the unreliability rate was 27% (there were 8 unreliable test results).

The randomness of the reliability problem made it very challenging to find out the exact software bug or hardware problem, that was causing it. **Figure 27** illustrates the existence of unreliable performance in each node. It shows that the problems existed at random nodes no matter if the node was the Gateway Node or the Communication Node. Moreover, the software upsets existed at random moments of time. It was noticed that resetting the Radio Frequency Controller (RFC) resumed the node back to normal operation. This indicated that there was a software bug that was located to the RFC software.

In order to achieve better performance, the watch dog timer was enabled which was supposed to initiate reset function of RFC software when a node suffers from software upset. However, it did not operate as well as we expected. Later on the developers of the UWASA Node found out that there was a software bug in the RFC software implementation. The problem was located in Linker Configuration and one associated function. The problems with the wireless data logging system reliability disappeared after this bug was fixed, and the reliable performance was achieved.

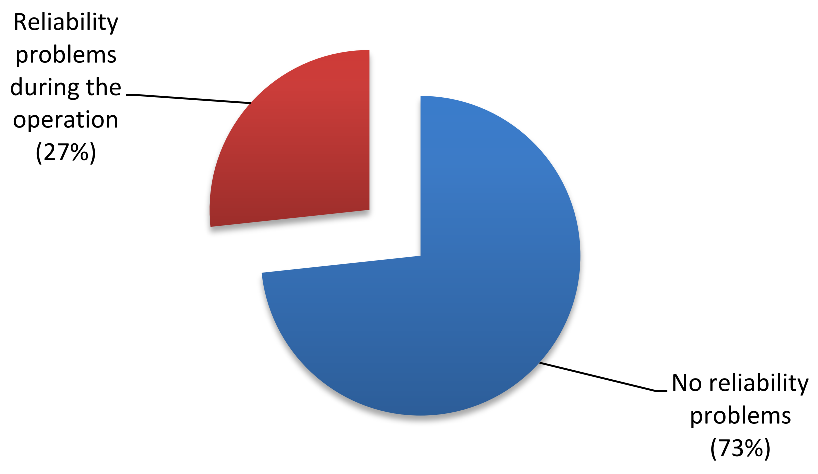


Figure 26. Data logger system operational reliability before the RFC software bug was fixed.

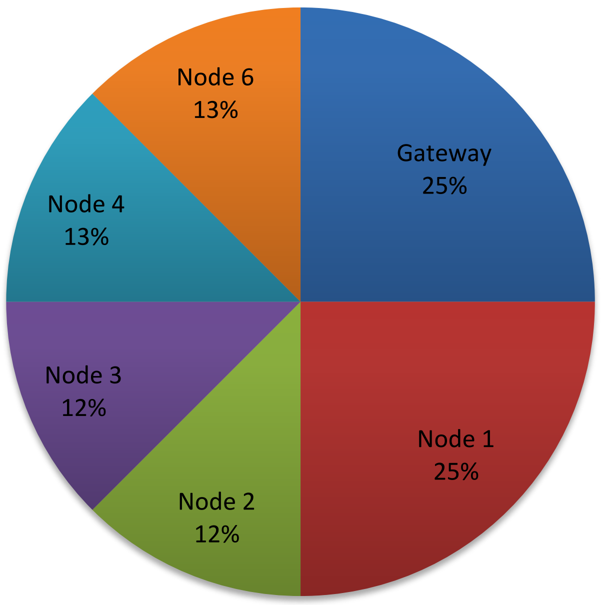


Figure 27. The distribution of performance problems before the RFC software bug was fixed.

7. CONCLUSIONS AND FUTURE WORK

7.1. Summary

In this thesis work, a WSN based wireless data logging system has been developed. The main goal of this thesis work was to collect user selected parameter's data from frequency converters and further transmit them to data logging PC through Wireless Sensor Network. This was enabled by connecting the UWASA Node with Vacon frequency converter and by developing a communication protocol for data collection over WSN. The motivation behind this application is to use collected data for analyzing of the behavior of a crane. This information can be further utilized in crane condition monitoring and control.

The concept of wireless automation is briefly discussed in the literature survey part. Several different types of WSN technologies have been introduced and their features have also been discussed. A wireless communication protocol has been designed for differentiating wireless messages from each other. A Graphical User Interface (GUI) was designed for the system configuring and testing.

According to experiment outcomes, the wireless data logging system is capable to collect maximum 10 parameters value from 6 frequency converters with a minimum logging interval of 200 milliseconds. At the beginning, the data logging system was suffering from a certain level of packet loss and unreliability. The fix of the antenna connections and the implementation of a resend mechanism improved significantly the system performance by reducing the packet loss. Later, a great reliability of the system was achieved by fixing a software bug that was pre-existing in the Radio Frequency Controller software. The software improvement was done only in the laboratory environment at the University of Vaasa, which suggests to perform further experiments in the industrial environment.

7.2. Future Work

In this work a communication between the UWASA Node and Vacon frequency converter was enabled. In the data logger application it was only used to collect data from the frequency converters for monitoring purposes. However, the implementation can also be used to send data to the frequency converters in a wireless manner. Various types of controlling commands can be sent to frequency converters over WSN.

The wireless transmission range in the data logger application varied between 0 and 30 meters. The communication range can be extended by applying multi-hop sensor network architecture. However, one should notice that the implementation of the multi hop technology will increase the routing complexity. In addition, some extra communication delay will be caused by the multi-hop relaying mechanism.

REFERENCES

Ahmad Kaleem, Stefan Heiss & Uwe Meier (2008). *Wireless Automation Systems Optimizing Reliability, Security and Coexistence*. Germany: Hochschule Ostwestfalen-Lippe University of Applied Sciences. Available from the Internet: < http://www.hs-owl.de/init/uploads/tx_initdb/Wireless-Automation-Users-Guide-final-sec.pdf>.

Contiki OS (2014). *Introduction of Contiki OS*. Available from the Internet: < http://www.contiki-os.org>.

Decotignie Jean-Dominique (2002). *Wireless Fieldbusses - A Survey of Issue and Solutions*. Barcelona, Spain: 15th Triennial World Congress, IFAC. Available from the Internet: < http://www.nt.ntnu.no/users/skoge/prost/proceedings/ifac2002/data/content/02039/2039.pdf >.

FreeRTOS (2014). *Introduction of FreeRTOS*. Available from the Internet: <http://www.freertos.org/>.

Ikram Waqas & Thornhill Nina F (2010). *Wireless Communication in Process Automation: A Survey of Opportunities, Requirements, Concerns and Challenges*. Coventry, United Kingdom: UKACC International Conference. E-ISBN 978-1-84600-038-6.

Iiro Jantunen (2008). Introduction to Wireless Automation - *Wireless Communication Methods for Sensor Networks*. 1st Ed. Helsinki University of Technology. 193p. ISBN 978-951-22-9370-4.

KONECRANES (2012). *Remote Monitoring and Reporting*. Available from the Internet: <http://www.konecranes.com/sites/default/files/download/konecranes-truconnect-remote-services-remote-monitoring-and-reporting.pdf>.

Kerovuori Juhani & Virrankoski Reino (2012). *RIWA Project Document for Wireless Data Logger for Frequency Converters*.

LAN/MAN Standards Committee (2003). *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*. New York, USA. ISBN 0-7381-3688-7.

Libelium World (2012). *Internet of Things Starter Kit*. [online] [cited 21st Oct 2013]. Available from the Internet: <http://www.libelium.com/ibm-and-libelium-launch-6lowpan-development-platform-for-the-internet-of-things/>.

MOS (2014). *Introduction of MANTIS Operating System*. Available from the Internet: < http://mantisos.org/index/tiki-index.php%3Fpage=HomePage.html >.

Mikael Pohjola (2008). *Introduction to Wireless Automation: Qos, Security and Energy Requirements in Wireless Automation Networks*. 1st Ed. Helsinki University of Technology. 193p. ISBN 978-951-22-9370-4.

Persson Lars (2007). *A Comparison between Fieldbuses and I/O for Instruments in the Process Industry*. Available from the Internet: <http://epubl.luth.se/1402-1617/2007/010/LTU-EX-07010-SE.pdf >.

Hoefel Roger Pierre Fabris (2014). *IEEE 802.11ac: A Performance Evaluation with Lattice-based MMSE and Zero Forcing MIMO OFDM Receivers*. Washington, DC, USA: Wireless Telecommunications Symposium, 2014.

Sen Dipankar, Sen Prosenjit & Das Anand M. (2009). *RFID for Energy and Utility Industries.* PennWell. ISBN 978-1-59370-105-5. pp. 1-48.

Sikora Axel (2009). *Wireless for Industrial and Process Automation - Trends, Challenges and Protocols*. Available from the Internet: <http://www.ba-loerrach.de>.

Schultz Stephan (2007). *Wireless Goes Process Automation – Challenges in Hazardous Areas*. Paris: PCIC Europe 2007. ISBN 978-3-9523333-0-3.

TinyOS (2014). *Introduction of TinyOS.* Available from the Internet: < http://www.tinyos.net>.

Upton Liz (2012). *Made in the UK.* Available from the Internet: < http://www.raspberrypi.org/made-in-the-uk/>.

Vieira, Marcos A.M., Adriano B. da Cunha & Diogenes C. da Silva Jr. (2006). *Designing Wireless Sensor Nodes*. Federal University of Minas Gerais. ISBN 0302-9743 3-540-36410-2.

Virrankoski Reino (2012). *Generic Sensor Network Architecture for Wireless Automation*. 1st Ed. University of Vaasa, Finland.

Yiğitler Huseyin, Reino Virrankoski & Mohammed Elmusrati (2010b). *Stackable Wireless Sensor and Actuator Network Platform for Wireless Automation: The UWASA Node*. In: Aalto University Workshop on Wireless Sensor Systems [online]. Aalto University Wireless Systems Group. Helsinki: Aalto University.

Yiğitler Huseyin (2012). *The UWASA Node Reference Manual 3.0.0*. 3rd Ed. Helsinki, Finland: Aalto University, 2012. Available from the Internet: <URL: http://wsn.ttk.fi/en/software/uwasa_manual.pdf>.

ZigBee Alliance (2014). *Introduction of ZigBee Alliance*. Available from the Internet: <http://www.ZigBee.org/About/UnderstandingZigBee.aspx>, <http://www.zigbee.org/Specifications/ZigBee/FAQ.aspx#8>.

Zigbee Alliance (2008). *Zigbee Specification, Document 053474r17*. Available from the Internet:<http://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2011/kjb79_ajm232/pmeter/ZigBee%20Specification.pdf>.

APPENDIXES

APPENDIX 1. Communication Node - Gateway link packet structures.

Table 1.1. Node Check packet from Gateway Node to Communication Node 1.

Destination	CS	Data	CRC
0x0001	0x01	NULL	1 byte

Table 1.2. The Acknowledgement packet from Communication Node 1 to Gateway Node for receiving Node Check packet.

Destination	CS	Data	CRC
0x0000	0x01	0x06 (ACK)	1 byte

Table 1.3. The Acknowledgement packet from Communication Node to Gateway Node for receiving Configuration Command.

Destination	CS	Data	CRC
0x0000	0x02	0x06 (ACK)	1 byte

Table 1.4. Read Command from Gateway Node to Communication Node.

Destination	CS	Data	CRC
0xFFFF (broadcast)	0x05	Sequence Number (4 bytes)	1 byte

Table 1.5. Response packet from Communication Node to Gateway Node.

Node ID	**CS**	**Data**								**CRC**
		Sequence number	Read command identifier 1	Memory index/ ID 1	Data from index 1	…	Read command identifier 10	Memory index/ID 10	Data from index 10	
0x0000	0x06	4 byte	1 bytes	2 bytes	4 Bytes		1 bytes	2 bytes	4 Bytes	1 byte

Table 1.6. Response Request packet from Gateway Node to Communication Node.

Destination	**CS**	**CRC**
2 bytes	0x06	1 byte

Table 1.7. Node addresses

Gateway	0x0000
Node 1	0x0001
Node 2	0x0002
Node 3	0x0003
Node 4	0x0004
Node 5	0x0005
Node 6	0x0006
Broadcast	0xffff

APPENDIX 2. Gateway - Data Logging PC link packet structures.

Table 2.1. The structure of Stop Command from Data Logging PC to Gateway Node.

Start	Length	Node ID	CS	Data	End or CRC
				Stop command	
0xAA	1 byte	0xFFFF Broadcast ID	0x04	0x00	0x03 (ETX=)

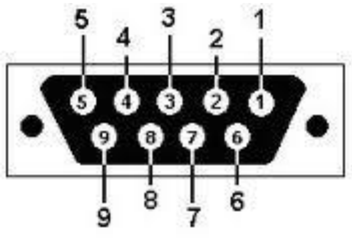
Table 2.2. Read command from Data Logging PC to Gateway Node.

Start	Length	Node ID	CS	End or CRC
0xAA	0x31	2 byte	0x05	0x03

Table 2.3. The data packet from Gateway Node to Data Logging PC.

Start	Length	Node ID	CS	Data								End or CRC
				Read command identifier 1	Memory index/ ID 1	Data from index 1	…	Read command identifier 1	Memory index/ID 10	Data from index 10	Time stamp	
0xAA	74 bytes	2 byte	0x06	1 bytes	2 bytes	4 Bytes		1 bytes	2 bytes	4 Bytes	4 byte	0x03 (ETX=)

APPENDIX 3. Pin-out of the D2V front panel HMI connector.



Pin number	
2	Receive data
3	Transmit data
5	Signal ground
6	+10V power supply