UNIVERSITY OF VAASA

FACULTY OF TECHNOLOGY

COMUNICATIONS AND SYSTEMS ENGINEERING

Le Manh Linh

WIRELESS SENSOR NETWORK INTEGRATION WITH FREQUENCY CONVERTER BY USING MULTI-HOP PROTOCOL

Master's thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, February 12, 2015.

Supervisor Mohammed Salem Elmusrati

Instructor Reino Virrankoski

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Professor Mohammed Salem Elmusrati and Senior Researcher Reino Virrankoski for their elaborate guidance and constant support throughout the whole work in this thesis.

Secondly, I am thankful to my working group Markus Madetoja, Maiwulan Wulayinjiang, Caner Cuhac and other people for helping and providing the required tools of the project.

Finally, I thanks to my family and friends for their endless encouragement. Without your mental supports, this Thesis would not be completed.

LE MANH LINH

Vaasa, Finland, February 12, 2015

Table of Contents

LIST OF FIGURES AND TABLES

SYMBOLS AND ABBREVIATION

ACK	Acknowledgement
ADC	Analog-to-digital Converter
CS	Carrier Sense
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CRC	Cyclic Redundancy Check
DAC	Digital-to-Analog Converter
DMA	Direct Memory Access
FFD	Full-function Device
FIFO	First In First Out
GPIO	General Purpose Input/Output
GW	Gateway
IEEE	Institute of Electrical and Electronics Engineers
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
JTAG	Joint Test Action Group
LAN	Local Area Network
LR-WPANs	Low-Rate Wireless Personal Area Networks
MAC	Medium Access Control
MC	Main Controller

MCU	Microprocessor Control Unit
MLME	MAC Layer Management Entity
MPDUs	MAC protocol Data Unit
MUTEXES	Mutual Exclusion
OS	Operating System
OSI	Open System Interconnection
PAN	Personal Area Network
PHY	Physical
PWM	Pulse-Width Modulation
RF	Radio Frequency
RFC	Radio Frequency Controller
RFD	Reduced-function Devices
RS-232	Recommended Standard 232
RTC	Real-Time Clock
RTOS	Real-Time Operation System
RX	Receiver
SPI	Serial Peripheral Interface
SoC	System on Chip
SSP	Synchronous Serial Port
SYNC	Synchronization
TCP	Transmission Control Protocol

ToP	Type of Packet
TX	Transmitter
UART	Universal Asynchronous Receiver/Transmitter
USART	Universal Synchronous Receiver/Transmitter
USB	Universal Serial Bus
WDT	Watchdog Timer
WSN	Wireless Sensor Network
WPAN	Wireless Personal Area Network

UNIVERSITY OF VAASA

Faculty of Technology

Author:	Le Manh Linh
Topic of Thesis:	Wireless Sensor Network Integration with Frequency Converter by Using Multi-Hop Protocol
Supervisor:	Mohammed Salem Elmusrati
Instructor:	Reino Virrankoski
Degree:	Master of Science in Technology
Department:	Department of Computer Science
Degree Programme:	Degree Programme in Information Technology
Major of Subject:	Telecommunication Engineering
Year of Entering the University:	2012
Year of Completing the Thesis:	2014 Pages: 91

ABSTRACT:

The area of wireless sensor network is rapidly growing and a corresponding growth in the demand of its applications. The need to collect data from remote positions is in fact unavoidable. Since the reliable communication distance in WSNs is relatively short, a multi-hop communication must be applied to be able to cover larger areas. A generic, modular and stackable WSN node, named UWASA Node has been developed by the University of Vaasa and Aalto University. Besides, the UWASA Node has been interfaced to Vacon Frequency Converter to be able to collect the frequency converter data in a wireless manner. A multi-hop communication will remarkably increase the coverage of the WSN, which is used to collect the frequency converter data.

In this thesis, a multi-hop protocol for collecting data from frequency converter is implemented to UWASA Nodes. Firstly, an overview of IEEE 802.15.4 architecture and routing protocols are introduced. Then the basic of WSN platform of the UWASA Node is discussed. Finally, a design and implementation of multi-hop protocol is explained in detail, and several experiments are performed to verify the system performance under the protocol. Developed protocol provides a solution to collect data and control remote devices.

KEYWORDS: Wireless Sensor Network, UWASA Node, IEEE 802.15.4, Frequency Converter, Routing Protocol, Condition Monitoring

1. INTRODUCTION

The emerging field of WNSs combines sensing, computation and communication into a single tiny device. Recently, WSNs have been developing rapidly and they are applied in many fields as industrial, health care, computer science and so on. Generally, a wireless sensor node contains of at least a microprocessor, radio transceiver, memory, power source, analog-to-digital converter (ADC) and development board. WSN consists of a large number of wireless sensor nodes which can process data, gather information and communicate with each other within the network.

There are many kinds of WSN topologies, from which mesh topology is beneficial because it can support large networks. While the capabilities of any single device are minimal, the use of hundreds of connected devices offers completely new technology possibilities. One challenge in the mesh topology is to keep the network operating when some malfunctioning occurs such as a node stops working or a connection between two nodes breaks.

Recently, Communication and System Engineering Group in University of Vaasa, jointly with Aalto University, has created a wireless sensor platform targeted especially for wireless automation applications. The platform is called the UWASA Node, and its feasibility has been verified by building five different pilot applications in different areas: industrial environment, wind turbines, distributed energy production, greenhouse and cattle house. Additionally, the business potential of the system and the ways to commercialization were also highly considered in the research work. (Virrankoski 2012: 1.)

In this work, we utilize UWASA Nodes to collect data from frequency converters. We build a mesh network consisting of UWASA Nodes with three kinds of tasks: nodes which are connected to frequency converters are called the communication nodes, nodes operating in the network and supporting the multi-hop communication are called the network nodes and the node which acts as a gateway between the network and PC is

called the gateway node. All nodes are able to find the best way which is defined by some parameters like bandwidth, time delay, hop count and so on or the shortest path which simply is defined by how many hops between nodes to get their destination basing on their own routing table. Each node builds its own routing table every 5 minute. There are three main challenges to overcome in the system development: the first is how to construct the WSN of the system for the purpose of communication with frequency converters. The second is how the nodes can find the communication path to the desired destinations. The last one is how the schedule the routing table computation and updates.

This thesis is divided into seven chapters. In Chapter 2, IEEE 802.15.4 communication protocol is introduced. Chapter 3 discusses about wireless sensor nodes and gives a brief description of the UWASA Node. The UWASA Node integration with Vacon frequency converter is explained in Chapter 4. A multi-hop protocol, which is developed in this thesis work, is explained in detail in Chapter 5. The performance of the developed protocol is evaluated through experiments in Chapter 6. Finally conclusions and some directions to the future work are presented in Chapter 7.

2. WIRELESS SENSOR NETWORKS

This chapter introduces , a overview of wireless sensor network firstly. Three network topologies are presented in this chapter. Then, the background, basics of IEEE 802.15.4 architecture will be shown. In addition, in IEEE 802.15.4 architecture, there are various devices and each takes care of different task. Finally, this chapter explains two types of routing protocols which are two main routing protocols in IEEE 802.15.4 .

2.1. General

A wireless sensor network (WSN) of spatially distributed autonomous sensors to monitor physical or environment conditions, such as temperature, sound, pressure, etc.

and to cooperatively pass their data through the network to a main location. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer applications, such as process monitoring and control, machine health monitoring, and so on.

A wireless sensor network can contain from few up to hundreds or even thousands of nodes. Each node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting.

A wireless sensor network is typically self-organizing and self-healing. Self-organizing networks allow a new node to automatically join to the network without the need for manual intervention. Self-healing networks enable nodes to reconfigure their link associations and find alternative pathways around failed or powered-down nodes. The implementation of these properties is specific to the network management protocol and the network topology, and ultimately will determine the network's flexibility, scalability, cost and performance.

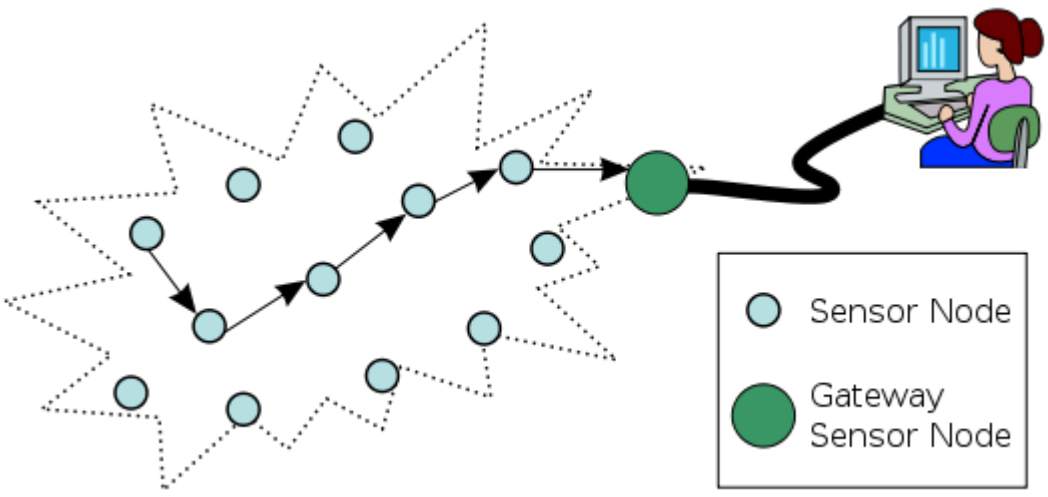


Figure 1. Wireless sensor network

The main characteristics of a WSN include: [Cuhac Caner, Huseyin Yigitler (2012).]

- Power consumption constraints for nodes using batteries or energy harvesting
- Ability to cope with node failures
- Mobility of nodes
- Heterogeneity of nodes
- Scalability to large scale of deployment
- Ability to withstand harsh environment conditions
- Ease of use
- Cross-layer design

2.2. IEEE 802.15.4 Communication Protocol

2.2.1. Overview of IEEE 802.15.4

The IEEE 802.15.4 standard was introduced by IEEE to fill a niche left by the existing wireless network standards which included IEEE 802.15.1 Bluetooth and 802.15.3 high-rate WPAN (Wireless Personal Area network). IEEE 802.15.4 is the standard that specifies the physical layer and media access control for low-rate wireless personal area networks. It focuses on the low cost communication of short-range devices with little underlying infrastructure, intending to reduce power consumption. There are a variety of application of IEEE 802.15.4 in many fields which are Home automation and security, consumer products, Health care, and Vehicle monitoring.

IEEE 802.15.4 was design to operate in unlicensed radio frequency bands which are not the same in all territories of the world. IEEE 802.15.4 consists three possible bands which are centered on the following frequencies: 868, 915 and 2400MHz. In which 868 MHz and 915 MHz bands have some advantages such as fewer users, less interference, and less absorption and reflection, but 2400 MHz band is far more widely adopted because of low power, highest data-rate (250 kbps) and worldwide availability for unlicensed use. IEEE 802.15.4 is able to avoid interference between radio

communications- that is, to select the best frequency channel and, where possible, to adapt to a changing RF environment by picking another channel if the current channel met a problem.

The range of a radio transmission is dependent on the operating environment. For example, with standard devices, at outside, a range can reach 200 meters and at inside, this can be reduced due to absorption, reflection, diffraction and standing wave effects caused by walls and other solid objects but typically a range of 30 meters can be reached. In addition, the range between devices can be extended in an IEEE 802.15.4-based network by employing a topology that uses intermediate nodes as stepping stones when passing data to the destination.

2.2.2. Network Topology

There is a variety of network topologies supported by IEEE 802.15.4. The main types of the supported network topologies are star, tree and mesh topology.

Star topology

Start topology is shown in Figure 2. This is basic type of network topology. Each node in the network is connected directly to the gateway node. This topology has the disadvantage if the link fails then there is no alternative route.

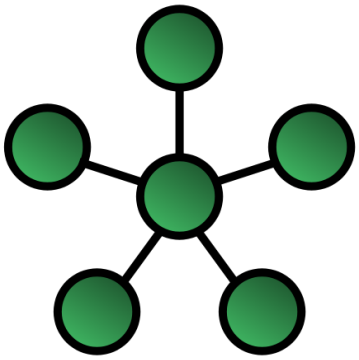


Figure 2. Star topology

Tree topology

Tree topology is based on parent-child architecture. Each node has its parent node and the node have one or more children. Each node can communicate only with its parent and its children.

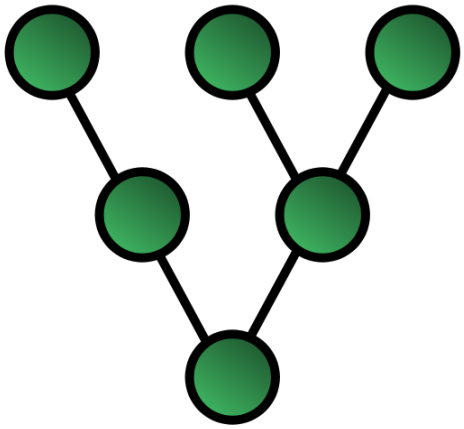


Figure 3. Tree topology.

Mesh topology

In the Mesh network topology, the devices can be identical and are deployed in an ad-hoc manner. The nodes connected to each other either by a single link or by a multi-hop path. One disadvantage in this sort if topology is that there can be several alternative routes between nodes so that if one node is lost, it can be compensated by another.

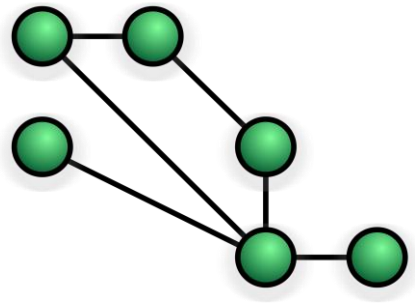


Figure 4. Mesh topology

2.2.3. Device Types.

The IEEE 802.15.4 standard defines two types of network devices: full-function device (FFD) and reduced-function devices (RFD). The first one can serve as the coordinator of a personal area network just as it may function as a common node. It implements a general model of communication which allows it to talk to any other device. The second one is a simple device with modest resource and communication requirements.

The network topology can be either peer-to-peer or star as illustrated in Figure 2. In the star topology the leaves on the outer edge are RFDs that can only communicate with their PAN coordinator node. In practice this means that the FFD checks the address of all incoming packets and discards all others except the packets originating from its PAN coordinator, since it can receive all packets from nodes inside its communication range. A RFD device in star topology cannot participate in routing, since it only communicates with its own PAN coordinator (IEEE 802.15.4, 2003)

Each device in an IEEE 802.15.4 network can have two types of addresses. The first one is IEEE (MAC) address which is a 64 bit address, allocated by the IEEE, which uniquely identifies the device. The second one is a 16 bit address, which can be given to the node when a node joins to the network.

IEEE 802.15.4 MAC layer is responsible for network establishing and providing links between MAC entities in different nodes. It also takes care of carrier sense multiple access collision avoidance (CSMA/CA) for channel access.

2.3. Routing Protocols

2.3.1. Distance Vector Routing Protocol

The term distance vector refers to the fact that the protocol manipulates vectors (arrays) of distance to other nodes in the network. The vector distance algorithm was the original ARPANET routing algorithm and also used in the internet under the name of RIP.

Device using distance-vector protocol do not have knowledge of the entire path to a destination. They are using two methods which are direction in which device or exit interface a packet should be forwarded and distance from its destination.

Distance-vector protocols are based on calculating the direction and distance to any link in a network. “Direction” means usually the next hop address and the exit interface. “Distance” is a measure of the cost to reach a certain node. The least cost route between nay two nodes is the route with minimum distance. Each node maintains a vector (table) of minimum distance to every node. The cost of reaching a destination is calculated using various route metrics.

Distance-vector routing algotrithm is a distributed algorithm or the Bellman-Ford algorithm. The Bellman-Ford algorithm is an algorithm that computes shortest paths from a single source vertex to all of the other vertices in a weighted digraph. It is slower than Dijkstra’s algorithm for the same problem, but more versatile, as itt is capable of handling graphs in which some of the edge weights are negative numbers. But the Bellman-Ford algorithm does not prevent routing loops from hapenning and suffers from the count-to-infinity problem. The count-to-infinity problem is that if A tells B that it has a path somewhere, there is no way for B to know of the path has B as a part of it.

There are characteristics of distance-vector routing protocol. The first characteristic is periodic updates which means that at the end of a certain time period, updates will be transmitted. This period typically ranges from 10 seconds to 90 seconds. At issue here is the fact that if updates are sent too frequently, congestion may occur, if updates are sent too infrequently, convergence time may be unacceptable high. The second one is neighbors which always means devices sharing a common data link. A distance vector routing protocol send its updates to neighboring routers and depends on them to pass the update information along to their neighbors. For this reason, distance vector routing is said to use hop-by-hop updates. The next one is Broadcast Updates. When a device first becomes active on a network, how does it find other devices and how does it announce its own presence? The simplest method is to send the updates to the broadcast address (255.255.255.255). neighboring devices speaking the same routing protocol will hear the broadcast and take appropriate action. Other devices uninterested in the routing updates will simply drop the packets. Most distance vector protocols take the every simple approach of telling their neighbors everything they know by broadcasting their entire route table. neighbors receiving these updates glean the information they need and discard everything else. Beside full routing table updat characteritic, trigged update is an important one of routing protocol which is knew as flash updates and very simple. If a metric changes for better or for worse, a router will immediately send out an update without waiting for its update time to expire. reconvergence will occor far more quickly than if every router had to wait for regularly scheduled updates and the problem of counting to infinity is greatly reduced, although not completely eliminated. Regular updates may still occur along with triggered updates. Thus a device might receive bad information about a route from a not-ye-reconverged device after having received correct information from a triggered update. The last one is Holddown timers. Triggered updates add responsivenness to a reconverging internetwork. Holddown timers introduce a certain amount of skepticism to reduce the acceptance of bad routing information. if the distance to a destination increases for example the hop count increases from 2 to 4, the device sets a holddown timer for that route. Until the timer expires, the router will not accept any new updates for the route. The holddown timers

must be set with care. If the holddown period is too short, it will be ineffective, and if it is too long, normal routing will be adversely affected.

2.3.2. Link State Routing Protocol

Link sate routing protocols are like a road map. A link state device cannot be fooled as easily into making bad routing decisions, because it has a complete picture of the network. The reason is that unlike the routing-by-rumor approach of distance vector, link state devices have firsthand information from all their peer devices. Each device originates information about itself, its directly connected links, and the state of those links. This information is passed around from device to device, each device making a copy of it, but never changing it. The ultimate objective is that every device has identical information about the internetwork, and each device will independently calculate its own best paths.

Link state protocols sometimes called shortest path first are built around the Dijkstra's shortest path algorithm. Each node periodically sends a short message, the link-state advertisement, which identifies the node which is producing it, all the other nodes to which it is directly connected and a sequence number, which increases every time the source node makes up a new version of the message.

The message is flooded throughout the network. As a necessary precursor, each node in the network remembers, for every other node in the network, the sequence number of the last link-state message which it received from that node.

Starting with the node which originally produced the message, it sends a copy to all of its neighbors. When a link state-sate advertisement is received at a node, the node looks up the sequence number it has stored for the source of that link-state message. If this message is newer, it is saved, and a copy is sent in turn to each of that node's neighbors. This procedure rapidly gets a copy of the latest version of each node's link-state advertisement to every node in the network. Network running link state algorithms can

be also segmented into hierarchies which limit the scope of route changes. These features mean that link state algorithms scale better to larger networks.

Finally, with the complete set of link state advertisements, each node produces the graph for the map of the network. The algorithm iterates over the collection of link-state advertisement; for each one, it makes links on the map of the network, from the node which sent that message, to all the nodes which that message indicates are neighbors of the sending node.

Each node independently runs an algorithm over the map to determine the shortest path from itself to every other node in the network, generally some variant of Dijkstra's algorithm is used. This is based around a link cost across each path which includes available bandwidth among other things. A node maintains two data structures: a tree containing nodes which are done and a list of candidates. The algorithm starts with both structure empty, it then adds to the first one the node itself. The algorithm repetitively does following: All neighbor nodes which are directly connected to the node are just added to the tree (excepting any nodes which are already in either the tree or the candidate list) and the rest are added to the second list, and each node in the candidate list is compared to each of the nodes already in the tree.

The candidate node which is closest to any of the nodes already in the tree is itself moved into the tree and attached to the appropriate neighbor node. When a node is moved from the candidate list into the tree, it is removed from the candidate list and is not considered in subsequent iterations of the algorithm.

3. SENSOR NODES

3.1. General Requirements

Hardware

Cheap cost and small size are some of the key target features of the wireless sensor nodes. The main components of the sensor node are microcontroller, transceiver,

external memory, power source and one or several sensors. The controller performs tasks, processes data and controls the operation of other components in the sensor node. While the most common controller type is a microcontroller, other alternatives that can be used as a controller are, for example, a general purpose desktop microprocessor, digital signal processors and FPGAs. A microcontroller is often used in many embedded systems such as sensor nodes because of its low cost, flexibility to connect to other devices, ease of programming and low power consumption.

Software

The energy resources of the sensor nodes are very limited, and they are the critical resource in terms if the network lifetime. WSNs are meant to be deployed in large numbers in various environments, including remote and hostile regions, where ad-hoc communication are one of the key properties. Hence, algorithms and protocols need to target to life maximization, robustness and fault tolerance, and self-configuration.

Operating systems

Operating systems for wireless sensor nodes are typically less complex then general-purpose operating systems because of more strongly resemble embedded systems for two reasons. First, compared to general purpose platforms, the WSNs have typically less tasks to execute. Second, the limited energy resources does not enable a use of complex operating system in the nodes.

3.2. The UWASA Node

UWASA node is jointly designed by university of Vaasa and Aalto university. It has a modular and stackable hardware architecture as shown in Figure 5. The architecture consists of at least two Modules: Power and Main Module. These two modules contain all properties like wireless communication interface, support for many peripheral

interfaces, basic processing and memory, power management and distribution interfaces. Moreover, UWASA Node is designed with some Slave Modules which implement the application dependent hardware. The signaling and supplies between the modules is transferred by means of hardware stack connector.

3.2.1. Main Module

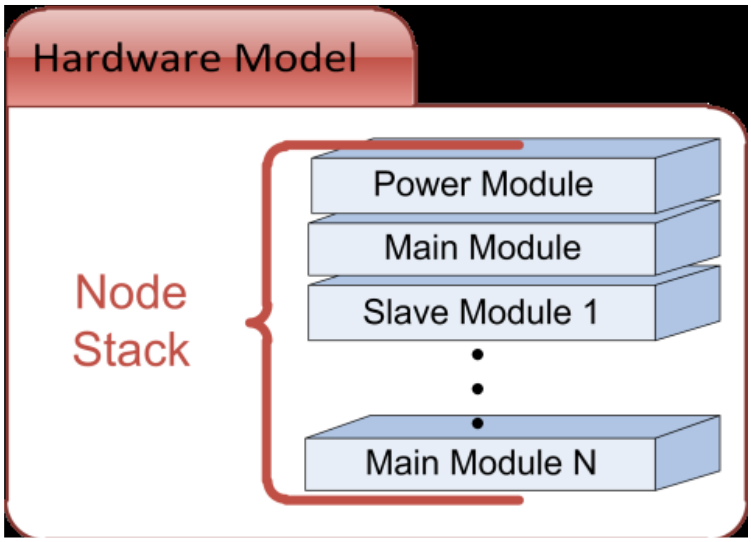


Figure 5. The hardware model of the UWASA Node [Cuhac Caner, Huseyin Yigitler (2012)]

The Main Module is the master module, which contains two processing units to allow transparent wireless communication while providing necessary means for stackable hardware architecture. The Main Module has three basic hardware blocks: Wireless Transceiver/RF controller, Main controller and MC-RFC interface circuit.

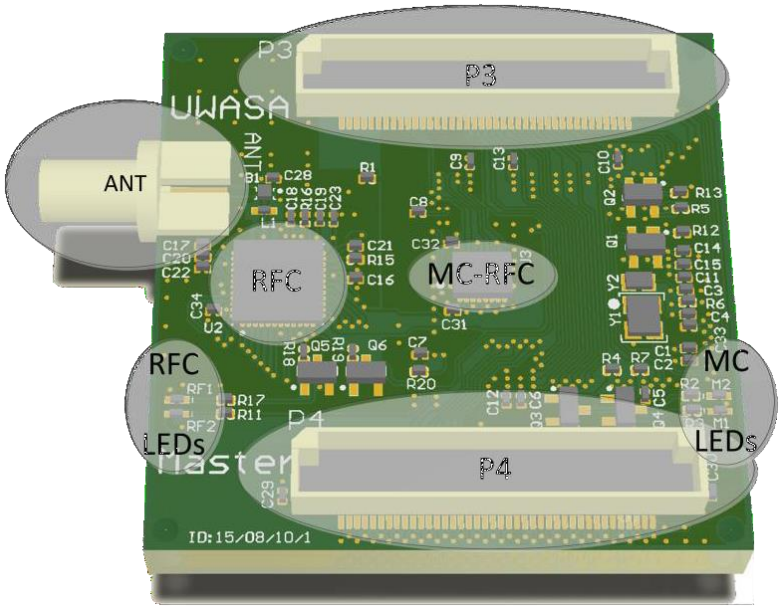


Figure 6. Top view of the Main Module. [Cuhac Caner, Huseyin Yigitler (2012)]

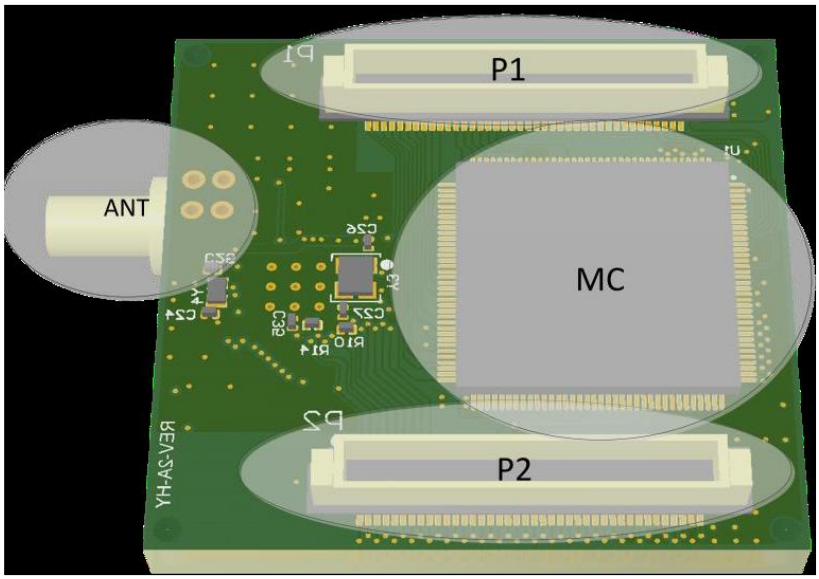


Figure 7. Top view of the Main Module. [Cuhac Caner, Huseyin Yigitler (2012)]

The Main Module performs all the main tasks, like wireless communication, managing in-node data exchange, performing data processing and decision making while supervising power interfaces and managing in-node power mode. The component block is presented in Figure 8.

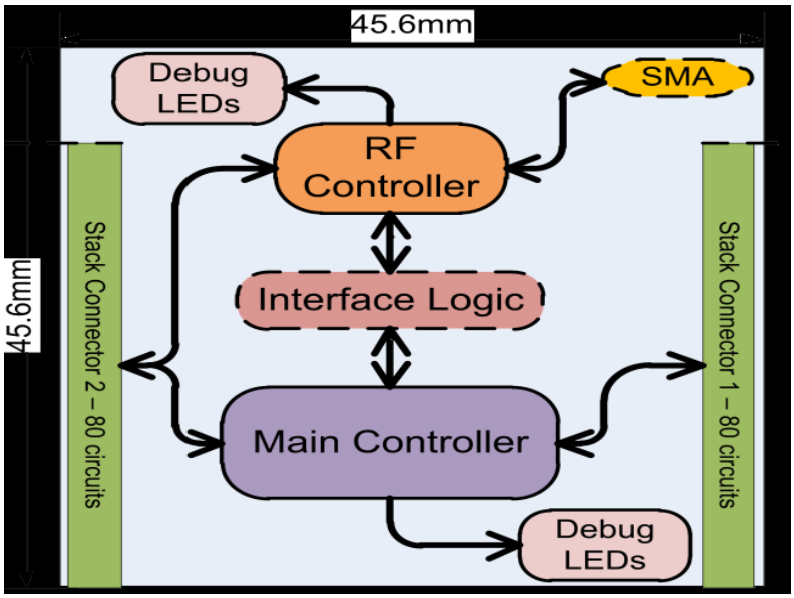


Figure 8. The component block of the Main Module. [Cuhac Caner, Huseyin Yigitler (2012)]

The main module has two controllers which are RFC and Main controller. RFC controller is CC2431 microcontroller and integrated IEEE 802.15.4 MAC hardware support and AES processor. RFC has 8051 based, 8 bit-processor that runs either 16MHz or 32Mhz, 128kB on-chip Flash and 8kB SRAM. Also, RFC contains of 21 GPIO Pins, 2 USARTs. 8 channel 10 bit Sigma-Delta ADC, 4 Times, Sleep Timer and 3 Power Modes.

Main Controller is the main processor of the Node which is LPC2378 from NXP. Main Controller has ARM7TDMI-S Pipelined processor 32bit processor that can run at up to 72MHz, 512kB on-chip Flash, 32kB SRAM on ARM local bus. EMC interface (16+1 bit). Main Controller has integrated Ethernet MAC, USB 2.0 full speed, SD/MMC

memory card interface, 4UART, 2 SSP (SPI), 2 I2C. 10 bit Sigma-Delta ADC, 4 Timers/Counters with 8 capture inputs/10 compare outputs. Moreover, integrated PLL, 2 independent power domains and 3 Power Modes belong to Main Controller.

RF Controller can switch off the power of Main Controller, when excessive process power is not needed; whereas the Main Controller can control enable states and voltage levels of power sources through the interfaces provided by the Power Module.

Main controller and RF front end of RF controller can disable individual peripherals and adjust operating frequencies. In addition, time synchronization is solved by the means of heart-beat signaling provided by either RF controller or Main Controller. RF Controller takes care of Network level time synchronization.

All debug features are on Development Board. RFC and MC have also debug LEDs on Main Module.

Main Controller

The Main Controller performs the main control of the Node. It is ARM7TDI-S with a lot of peripheral options.

Table 1. Description of Main Controller [Cuhac Caner, Huseyin Yigitler (2012)]

Processor	ARM7TDMI-S 32 bits Pipelined processor up to 72MHZ
Memory	512kB on-chip Flash 32kB SRAM on ARM local bus 16kb SRAM for Ethernet interface 8kB SRAM for GPDMA
Power	Models: Idle-Sleep-down Integrated PLL Peripheral Clock control
Clock	Main oscillator 12MHZ

	RTC Oscillator 32768Hz- Independent Supply
Debug Interface	Embedded ICE: JTAG - Embedded trace
Interfaces	104 GPIO Pins Ethernet MAC USB 2.0 Full Speed SD/MMC memory card interface 4 UART 2 CAN channels 2 SSP (1 of them can be SPI) 3 I2C I2S 8 Channels 10 bit Sigma-Delta ADC 10 bit DAC (1 channel) 4 Timer/Counter with 8 capture inputs/10 compare outputs
Power Consumption	125mA(max)
Supply Voltage	3.3 V
RTC Supply Voltage	2.0-3.6 V
VREF ADC Reference	3.0 V
Vih –Vil (high-low) Logical voltage levels	2.0-0.8 V
Vi Input Voltage Max	5V tolerant input (max 5.5 V)
Vo Output voltage Max	Vdd 3.3 V

RF Controller

The RF Controller is responsible for wireless communication function of the UWASA Node. It contains an integrated IEEE 802.15.4 MAC radio, and 8051 based 8 bit processor.

CC2431 System on Chip (SoC) of Texas Instruments is picked up as the RF controller.

Table 2. Description of RF Controller [Cuhac Caner, Huseyin Yigitler (2012)]

SoC Description	Integrated RF transceiver and microcontroller
Processor	8051 family 8 bit processor 32MHz processor
Memory	128kB on-chip Flash 8kB SRAM
Power	Model: idle-Sleep-power down Main clock selection Peripheral Clock control
Clock	Main Oscillator: either 32MHz ext OSC or 16MHz int RC OSC RTC Oscillator: ext 32768Hz or int 32kHz RC OSC
Interfaces	Integrated 2 IEEE 802.15.4 Digital Radio IEEE 802.15.4 MAC hardware support True random number Engine 21 GPIO pins 2 USARTs (UART and SPI) 8 Channels 10 bit Sigma-Delta ADC 4 timers: 1 general 16bit timer, 2 general 8bit timers, 1 MAC timer
Debug Interface	two wire debug interfaces
Power consumption	40mA(max)

Supply Voltage (VDD)	2.5V
Vih-Vil Logic voltage levels (high-low)	2.0-0.5 V
Vi Input voltage Max	0-VDD 2.5V
Vo output voltage max	VDD 2.5V

RF controller has three implemented functional blocks: MC Communication Block, MC control block, and Debug support block

MC-RFC Interface

Due different supply levels of main Controller and RF controller, a voltage level converter interface is required between them. Texas Instrument TXB0108 is selected as the bidirectional voltage level shifter with auto direction sensing due to its small form factor, low power consumption and high data rate support. The properties is shown below.

Table 3. Description of MC-RF Interface [Cuhac Caner, Huseyin Yigitler (2012)]

Port 1 Supply voltage, Vcc1	2.5 V
Port 2 Supply voltage, Vcc2	3.3 V
Data rate max	60 Mbps
power consumption	4uA (max)Icc

Form factor	QFN20 Package: 4.65x3.65x1 mm
Tristate pins	OE pin and Vcc isolation feature- all pins are put to high Z mode
OE pin	Port 1 controlled
Port 1 min High Voltage level, Voh1	Vcc1-0.4V
Port 2 min High Voltage level, Voh2	Vcc2-0.4V
Port 1 max low Voltage level, Vol1	0.4 V
Port 2 max low Voltage level, Vol2	0.4V

3.2.2. Development Board

In order to adapt many purposes of development, UWASA Node Main Module has the Development Board which aims to provide convenient development interfaces. Figure 9 and Figure 10 show all components blocks of the Development Board.

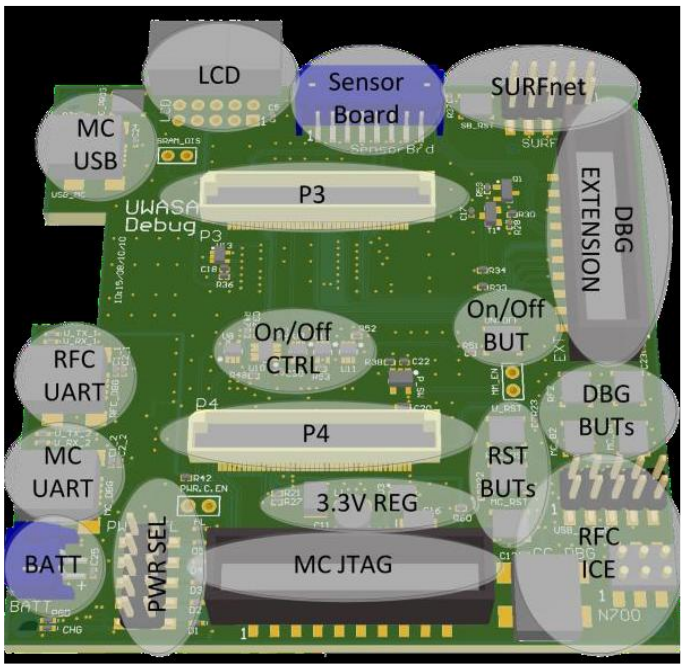


Figure 9. A top view of the UWASA Node Development Board. [Cuhac Caner, Huseyin Yigitler (2012)]

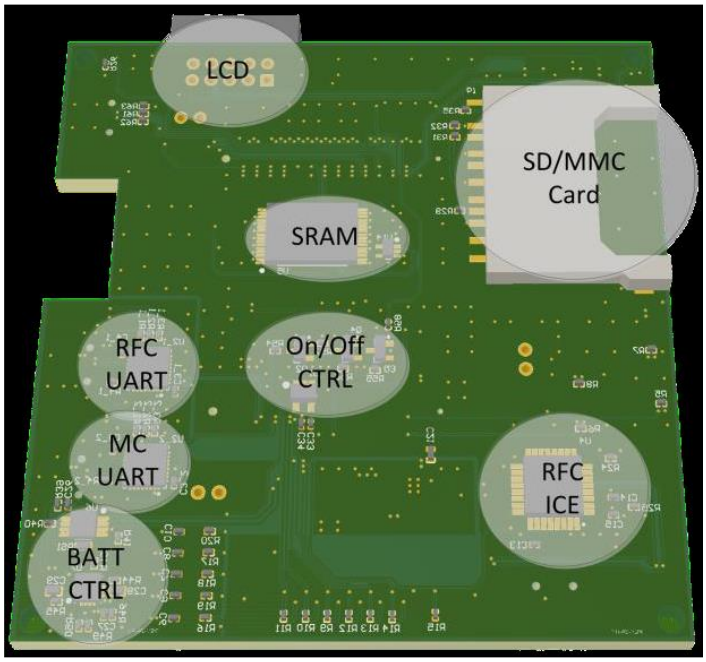


Figure 10. A Bottom view of the UWASA Node Development Board. [Cuhac Caner, Huseyin Yigitler (2012)]

There are two functional blocks of the Development Board:

- Main Module Debug Interfaces
- Development Extensions

Main Module Debug Interfaces are:

- MC JTAG connector
- RFC Debugger
- MC debug Buttons
- RFC Debug Buttons
- MC Debug UART-USB converter
- RFC Debug UART-USB converter
- MC In System programming (ISP)
- MC LCD connection

The Development Board has some development extensions to speed up application development and proof of concept prior to any hardware development. These interfaces are:

- SD/MMC card interface
- 128kByte SRAM connected to External Memory Controller of MC
- Sensinode N700 sensor debug interface connector
- MC USB interface implementation
- Battery management Block Implementation of Power Module 1
- MC RTC Alarm feature based power on7off control block implementation
- SURFnet Buttons Connector
- Sensor Board Connector
- Extension Connector

4. WSN INTEGRATION WITH VACON FREQUENCY CONVERTER

This chapter describes WSN based wireless data logger for collecting data from frequency converters. Figure 11 shows devices, communication links and the division of responsibilities. The network topology used in the system is star, where the gateway node schedules the transmission of the communication nodes. The communication nodes are attached to the frequency converters. Each one f them can communicate with the gateway node, but they cannot communicate with each other.

System is used for logging data in real time from several frequency converters. Two use cases are considered. In the first case, the system is set to monitor the usage of the crane for several days. Only changes are logged. The main focus is in this use case. In the second case, the system is used over a short periods of time for real-time data logging during the operation of the crane.

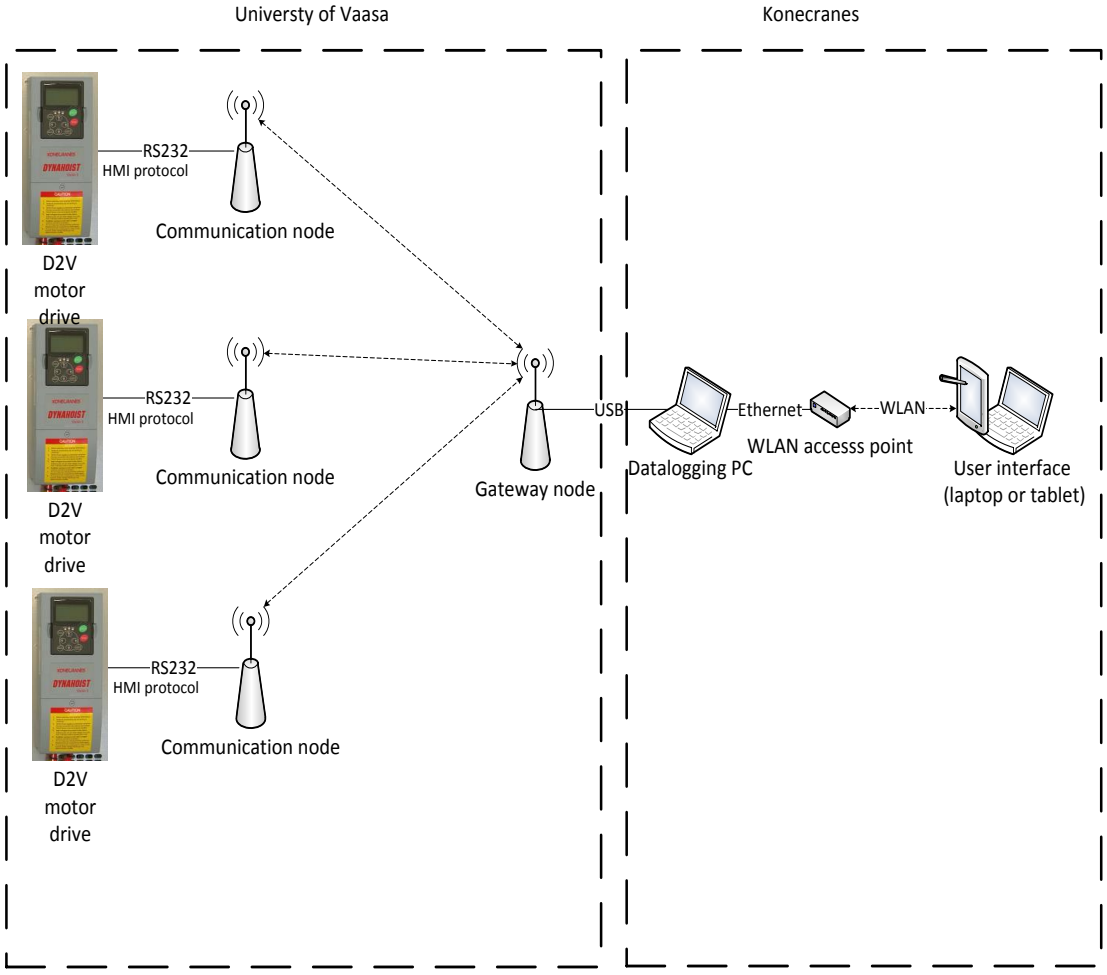


Figure 11. The system architecture.

The user can connect his own computer to the data logging PC via WLAN, and open the user interface software. Then the user can select up to ten frequency converter parameters to be logged. The data logging PC sends the parameter information to the Gateway Node, which transmits the configuration information to the Communication Nodes. The user can set the system to transmit the parameter values continuously with defined interval or, alternatively, only the changes in the measured data can be set to trigger logging. Logged data is saved to database which is located in the data logging PC. All data is time stamped and source node ID is linked with the data. Defined period of data can be loaded and plotted by using the user interface.

4.1. Communication Architecture

4.1.1. Communication between UWASA Node and Frequency Converter

The Communication Node is connected to the frequency converter over serial port (RS-232) and it communicate with the frequency converter by using the HMI protocol, which is already implemented in the UWASA node. Data can be requested either by index or by ID.

Frequency converter responses to read command presented in Table 4 by sending a response packet which is shown in Table 5. If more than 32 bytes are wanted to read at the same time, the frequency converter splits data into several packets. Hence, it makes the system possible to send 256 bytes at one time.

Table 4. Read by index command.

Start chars		command	Object	Read command identifier	index		number of variable		Stop char		CRC
DLE	STX	Read	variable object	Read by index	memory location				DLE	ETX	
0x01	0x02	0x40	0x0B	0x00	0x00	0xBC	0x00	0x01	0x01	0x03	0x66

Table 5. Response to single byte read commend

Start Chars		Command	Object	Data type	Data		Stop chars		CRC
DLE	STX	Response	Variable object	int	memory contents		DLE	ETX	
0x10	0x02	0xC0	0x0B	0x03	0x00	0x0	0x10	0x03	0x52

Frequency converter data can also be requested by ID. The data request by ID is presented in Table 6 and the frequency converter response in Table 7.

Table 6. Read by ID command

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	
0x01	0x02	0x40	0x0B	0x01	MSB	LSB	0x00	0x01	0x01	0x03	0x66

Table 7. Response to single read by ID command

Start chars		Command	Object	Data type	Data		Stop chars		CRC
DLE	STX	Response	Variable - object	int	memory contents		DLE	ETX	
0x10	0x02	0xC0	0x0B	0x03	0x00	0x00	0x10	0x03	0x52

Indexes depend on software versions, there are some samples of indexes in Table 8.

Table 8. Indexes, SW: PR03V130. (FCR-file from Vacon company)

Index	Type	Name
697	UNIT	User Password
732	UNIT	Acceleration Time
780	UNIT	Acceleration Time2

Table 9. Parameter IDs (FCR-file from Vacon company)

Parameter ID	Parameter Name
102	Acceleration Time
103	Deceleration Time
502	Acceleration Time2
503	Deceleration Time2

4.1.2. Communication in WSN .

The UWASA Nodes in the network are using current implemented RFC software to communicate with each other. IEEE address of the gateway is set to 0x0000. The address of Communication Nodes can be selected freely.

Gateway Node configures Communication Nodes by sending a configuration packet. ACK is requested to be sure that all communication nodes have received the packet. The configuration packet must include at least:

- Selection and setting of the interval based on the trigger based logging mode: Logging interval in case of continuous logging. Instruction on data transmission in case of trigged logging.
- Memory indexes for all logged parameters.
- Number of variables (how many bytes are read starting from given indexes)

When the gateway node starts the data logging, it sends a start command. The start command can be a broadcast packet, but ACK is requested from every node to be sure that all Communication Nodes have received the packet correctly. If ACK is not received by some of the Communication Nodes, Gateway node resends the packet. After 10 resends without ACKs, the communication link is marked broken and the resending is stopped. Stop data logging command packet acts otherwise similarly as start command, expect that when it is received by the Communication Nodes, the nodes stop requesting data from frequency converter. Once the Communication Node receives the requested data from the frequency converter, data packet sent contains at least following fields:

- Node ID (RF controller adds automatically)
- Logged memory index
- Logged data
- Time stamp or ID

4.1.3. Communicating between Gateway Node and Data Logging PC

This communicating link between Gateway Node and Data Logging PC is a virtual serial port over USB. Packet structure for communication between PC and Gateway Node is shown in Table 10.

Table 10. Packet structure for communication between PC and GW

Start char	Length	Node ID	Packet Type	Data	End char
0xAA	1 byte	2 bytes	1 bytes	n bytes	0x03

Length field in each packet indicates the length of the data part of each message. PC sends configuration, start, and stop commands. When start command is sent, the GW begins to send continuously data to PC. The packet types are presented in Table 11.

Table 11. Type of packet values

Type of packet values fro communication between PC and GW node	
0x01	check node
0x02	Configuration
0x03	Start
0x04	Stop
0x05	Read
0x06	Response

4.2. Devices in the Datalogger Setup

4.2.1. Frequency Converter

The frequency converters we apply are designed by Vacon company. Frequency converters are used to change the frequency and magnitude of the constant grid voltage

to a variable load voltage. Frequency converters are especially used in variable frequency AC motor control. Figure 12 shows a frequency converter.

UWASA Node and frequency converter can communicate each other by using frequency converter's VHML protocol which is also implemented to the UWASA Node, which is connected to the frequency converter. The VHML protocol defines a set of rules for communication between external devices and Vacon frequency converters.

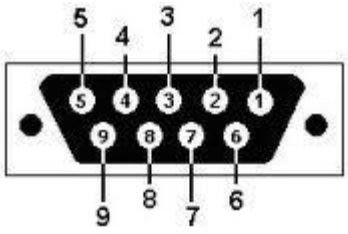
Figure 12.Frequency converter

4.2.2. Communication Node

Communication Node is UWASA node, which is connected to frequency converter via RS-232 serial port, is called Communication Node. Figure 13 shows the pin-out of the front panel connector. The frequency converter sends NULL packets approximately

once per second, if no traffic has been detected in the communication link. The NULL packets need to be ACKed, or otherwise the link is considered to be broken and the frequency converter does not send anything to the link before the communication node opens the connection again.

Figure 13. The third generation of UWASA Node working which is used as a Communication Node



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

Figure 14. The pin-out of the frequency converter connector.

4.2.3. Gateway Node

The UWASA Node, which is connected to the data logging PC via serial port, is called the Gateway Node. It communicates wirelessly with the Communication Nodes and over the serial port with the data logging PC. The second generation UWASA Node is used for this purpose.

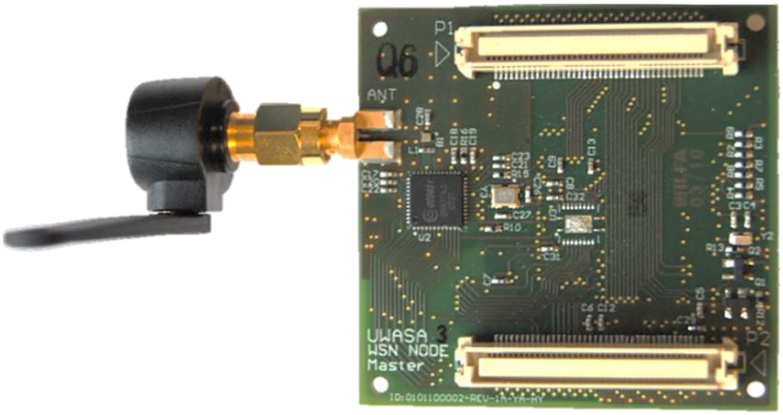


Figure 15. The second generation of UWASA Node, which is used as a Gateway Node

5. PROTOCOL FOR MULTI-HOP COMMUNICATION

In a multi-hop network, intermediate nodes have to relay packets from the source to the destination node. Such an intermediate node has to decide to which neighbor to forward an incoming packet not destined for itself. Typically, a routing table that lists the most appropriate neighbors for any given packet destination, are used. The construction and maintenance of these routing tables are the crucial tasks of a distributed routing protocol.

This chapter introduces a new multi-hop protocol for collecting data from remote frequency converters over wireless sensor network. The protocol takes advantage of distance. The single-hop protocol, that was applied to the same purpose, was just based on star topology and supported single-hop communication within a range of 10 meters. This chapter explains in detail the developed multi-hop protocol and how the gateway node can gather data from communication nodes and frequency converter. The collected data is then assed to from the Gateway Node to the data logging PC.

5.1. Designing a Multi-Hop Protocol

In the star topology setup, every Communication Node can communicate directly with the Gateway Node. The system communication and configuration protocol is quite simple. But the star topology system has always its limitations in terms of flexibility and reachable distance. With a single link communication between Gateway and Communication Nodes, the reliable communication range is about 10 meters.

The problems caused by single-hop setup can be solved by using multi-hop topology. The multi-hop protocol can support a large network up to 15 hops. Moreover, one of the advantage of the multi-hop protocol is that the node can choose a alternative path to reach its destination if a link in its previous path gets broken.

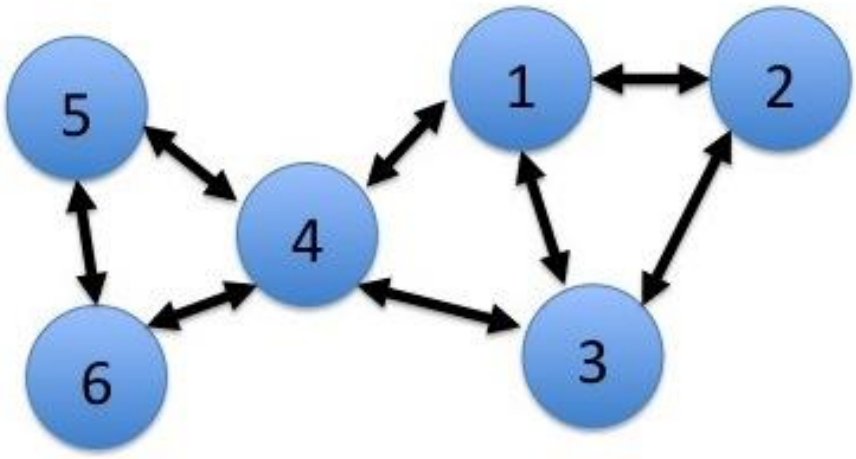


Figure 16. A simple example of routing in a multi-hop network

Whenever a source node cannot send its packets directly to its destination node but has to rely on the assistance of intermediate nodes to forward these packets on its behalf, a multi-hop network result- an example is shown in Figure 16. In such a network, an intermediate node (as well as the source node) has to decide to which neighboring node an incoming packet should be passed on so that it eventually reaches its destination. For example, assume that in the setup illustrated in Figure 16, the Node 1 wants to send a packet to Node 6. Node 1would not send packet to Node 2, if there exists a direct link between Node 1 and Node 4. This operation of passing is called forwarding, and there exists several options to organize the forwarding process.

The simplest way to forward an incoming packet is to broadcast it to all neighbors. As long as the source and destination nodes cannot communicate directly with each other or do not know which pathway to pass, it is unsure does the packet reach its destination. To avoid packet circulating endlessly in the network, a node should only forward packets it has not yet seen. Also, every packet usually contains of a field of expiration time to avoid needless propagation of the packet.

An alternative to forwarding the packet to all neighbors is to forward it to an arbitrary one. Such gossiping results in the packet randomly traversing the network in the hope of eventually finding the destination node. Clearly, the packet delay can be larger

compared to broadcasting. If there simple forwarding rules are applied, the overhearing (number of unnecessarily sent packets) and the transmission delay are usually large. Hence, much more efficient routing protocol can be developed, if we would have some information about the most suitable neighbors in terms of routing the packet towards the destination. A neighbor's suitability is based on the cost it takes to send a packet towards its destination via this particular neighbor. These costs can be measures in various metrics, for example, the minimal number of hops. Each node collects these costs in its own routing table.

Determining these routing tables is the task of the routing algorithm, which is implemented as a routing protocol. The routing table contains of IP address of nodes, metric as hop count and entry node where the node sends packet through. The protocol uses a hop count as a metric to measure the distance between the source node and the destination node. Each hop in a path from source to destination is assigned a hop count value, which is typically 1. When a packet passed through a node, hop count value is increased by 1. The maximum length of the path, the protocol can support is 15 hops. If hop count value becomes 15 and the receiving node is not the destination, it considers that packet to be never able to reach its destination and then removes.

Table 12. Routing table structure

Node Address	Hop count	Entry
0x0001	1	0x0001
0x0002	2	0x0001

5.2. Packet Types in the Multi-Hop Protocol

There are three types of packet in the multi-hop routing protocol.

Table 13. Packet types in the protocol

Type of packet	Number	Task
Requesting	0x01	Discovering a node's neighbors.
ACK packet	0x01	Confirming to the Source Node
Requesting routing table	0x08	Requesting a node's neighbors send their own routing table
Routing table update	0x07	Contains of a node's routing table.

Table 14. Requesting packet structure

Node Address	Packet type	Hop count	CRC
2 bytes	0x01	0x00	1 byte

The packet includes 4 fields: the address of the transmitting node, type of packet, number of hops the packet has been transmitted and checksum byte which is computed from an arbitrary block of digital data for the purpose of detecting errors which may have been introduced during its transmission or storage. The Hop count field is initialized of 0 and when goes through one node, it is increased by 1. The last field is checksum byte which is used in sender as well as receiver. This protocol apply a simple checksum algorithm which is so-called longitudinal parity check. It breaks the data into sequences with a fixed number of bits, and then computes the exclusive or of all those words. The result is appended to the message as an extra word. To check the integrity of a message, the receiver computes the exclusive or of all its words, including checksum; if the result is not zero, the receiver knows a transmission error occurred.

Table 15. The requesting routing table packet structure

Node Address	Packet type	Neighbor Address	Neighbor Address	…	CRC
2 bytes	0x07	1 byte	1 byte	1 byte	1 byte

The routing table packet contains the addresses of single hop neighbor of the one sending this packet. Routing table update has the same structure with the requesting routing table packet.

5.3. How Does the Multi-Hop Protocol Work?

The multi-hop protocol uses broadcast data packets to exchange routing information and to built routing table. Once the protocol is applied, the node is able to broadcast requesting packet once in 60 seconds, which is termed advertising. All nodes in the network can exchange messages in order to build routing tables. But only Gateway Node has a whole story of the network. Figure 17 illustrates a mesh topology, let's assume Node 1 is a Gateway Node and others are Communication Nodes and see what happening in this network.

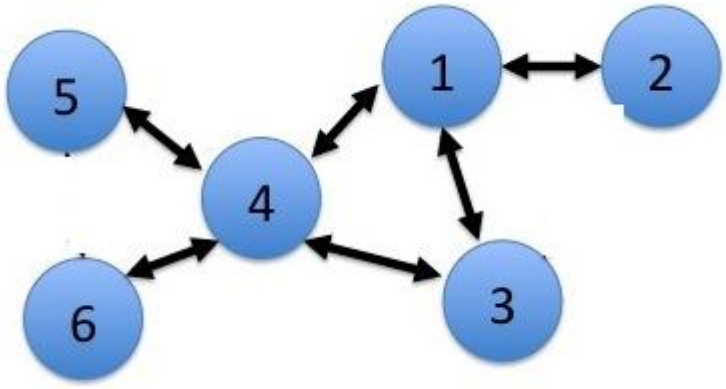


Figure 17. A mesh wireless network

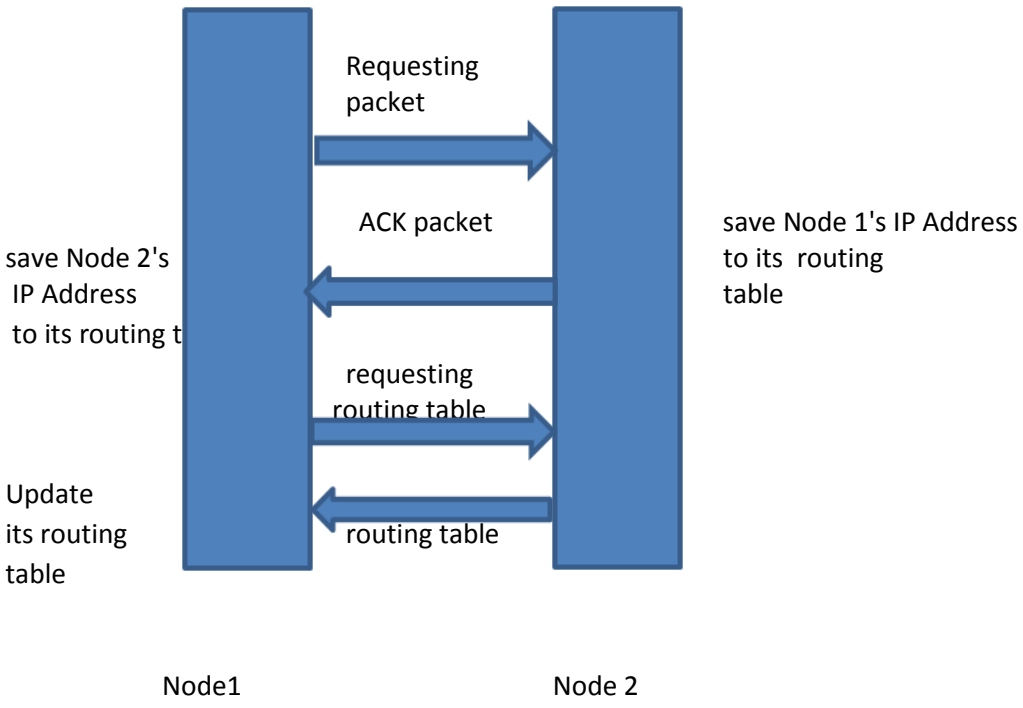


Figure 18. Two nodes exchanges messages

Figure 18 illustrates a communication session between two nodes. At the beginning, Node 1 has no idea about the network, so Node 1 will broadcast a requesting packet and wait for confirmation from nodes who have been received that packet. Node 2 received the requesting packet from Node 1 and saves Node 1's IP address to its own routing table. After that Node 2 sends back an ACK packet to Node 1 in order to inform that it received the requesting packet. Node 1 now adds Node 2's IP address to its routing table. When timeout for requesting single-hop neighbor is over, Node 1 starts to send requesting routing table packet to Node 2. Then as soon as receiving that packet Node 2 responses a routing table packet which contains its own routing table to Node 1. Thus, Node 1 has the whole story of Node 2 and Node 2 knows all single-hop neighbors of Node 1. That is a communication session between two nodes to build their routing table.

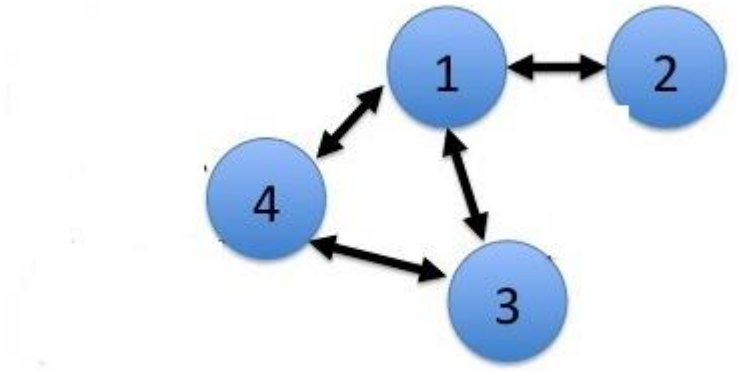


Figure 19. A small part of the network with Gateway Node.

In order to get more understandable about the protocol, let's divide the network into two parts. In the first part showed in Figure 19, Node 1 as Gateway has no idea about the network, it will broadcast a requesting packet to discover single-hop neighbors. In this case, three nodes can receive the requesting packet are Node 2, 3 and 4. They save Node 1's IP address to their own routing table and send back ACK packets to Node 1 separately. The timeslot for sending back the ACK packet is divided into three parts, each part is 100 ms in order to avoid conflicts. Node 1 receives ACK packets from three node one by one and save their IP addresses to the routing table. When timeout for collecting ACK packets is over, Node 1 sends requesting routing table to Node 2,3,and 4 respectively.

Node 2 receives the requesting table of Node 1, so it will get information about Node 1's single-hop neighbors firstly (here are Node 3 and 4) and then broadcast its own requesting packet to find out who are its single-hop neighbor. As the network topology, Node 2 has only connection with Node 1 but Node 1 will ignore Node 2's requesting packet because Node 1 received the ACK packet of Node 2 once. After Node 2 receives requesting routing table packet from Node 1, it sends back its own routing table which contains only Node 1's address to Node 1.

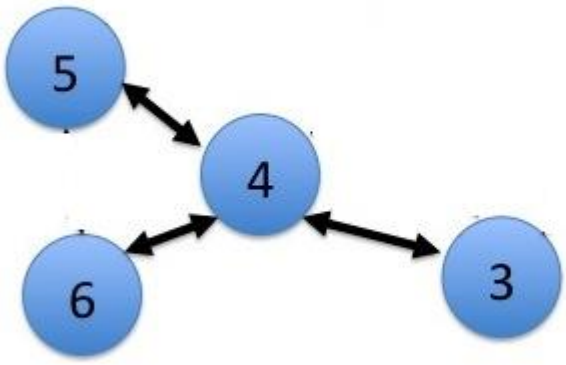


Figure 20. The another part of the network

Figure 20 shows 4 nodes connection which is Node 3, 4, 5, and 6. Node 3 has also connection with Node 4. So, Node 4 will receive Node 3's requesting packet and send back an ACK packet. After receiving a requesting routing table packet from Node 3, Node 4 sends back its own routing table which contains two other nodes Node 5 and 6. Now Node 3 has the whole story of Node 4, that means Node 3 can communicate with Node 5 or 6 through Node 4. Then Node 3 sends its final routing table to Node 1. Node 1 got knowledge about Node 3's connections. Node 1 knows the way to reach Node 5 or 6 is going through Node 3.

Node 4 broadcasts a requesting packet after receiving the requesting routing table packet from Node 1. That means it will not broadcast any requesting packet when it also receives the requesting packet for Node 3 later. Each node can broadcast a requesting packet one time in maximum. But it will send back its ACK and final routing table to Node 1 as soon as it receive the requesting packet and requesting routing table packet from Node 1 respectively. Node 1 now has all routing information of the network.

Table 16. The routing table of Node 1

Node Address	Hop Count	Entry
0x0002	1	0x0002
0x0003	1	0x0003
0x0004	1	0x0004
0x0005	2	0x0004
0x006	2	0x004

Table 17. The routing table of Node 4

Node Address	Hop count	Entry
0x0001	1	0x0001
0x0002	2	0x0001
0x0003	1	0x0003
0x0005	1	0x0005
0x0006	1	0x0006

Table 18. The routing table of Node 3

Node Address	Hop count	Entry
0x0001	1	0x0001
0x0002	2	0x0001
0x0004	1	0x0004

0x0005	2	0x0004
0x0006	2	0x0006

.

5.4. Implementing The Multi-Hop Protocol to the System

5.4.1. Implementing the Multi-Hop Protocol

Gateway Node

A Gateway Node is the one which is connected to data logging PC or to other gateway nodes of other networks. In communication with data logging PC through USB port and wirelessly with other nodes in the wireless sensor network.

Gateway Node's task is to initialize the routing protocol (requesting packet) and to set up timer for regular updating. A flowchart, that presents the Gateway Node operation, is presented in Figure 21.

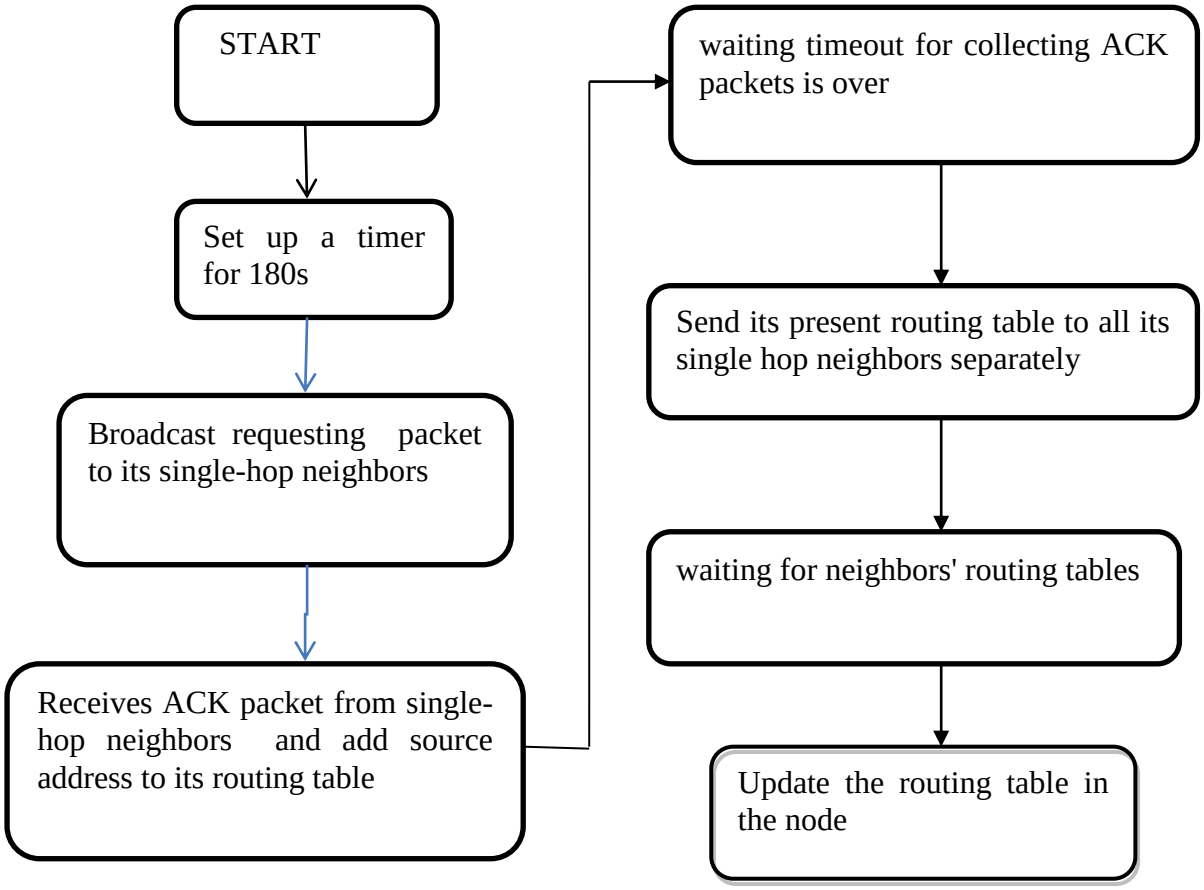


Figure 21. Flowchart of routing protocol operation in the Gateway Node

At the beginning, the gateway node sets up a timer to count 180s for routing process. It broadcasts a requesting packet in order to discover all single-hop neighbors. The requesting packet is used to discover all neighbor of the node. All nodes can connect directly with the node will receive that packet.

After broadcasting he requesting packet to all its single-hop neighbors, Gateway Node receives ACK packet from its neighbors separately and adds their address to its routing table. The ACK packet has same structure with requesting packet. The Gateway Node never receives ACK packet or requesting packet of a single node two times. After collecting all ACK packets from its neighbors, the Gateway Node sends routing table packet to its all neighbors separately.

After broadcast its own routing table, the gateway waits the routing table packets from neighbors. Since the Gateway Node received neighbors' routing table packet, it will compare them with its own in order to update its routing table. When the routing table is finished, the Gateway Node will use it to communicate in the network.

Communication Software

The Communication Node is able to connect with other devices, for example frequency Convertor, or motor and communicate with other Communication Nodes over wireless.

First of all, the Communication Node will wait for the initial requesting packet until the packet reached it. After receiving that packet successfully, the node sends back a confirmed packet to the initial node. Then, the node broadcasts a requesting packet and comes back to wait for other requesting packets if only if it received successfully a requesting routing table packet from another. If there are no new nodes coming, the node will send its own routing table to the node named "initial node". Otherwise, the Communication Node sends separately to all the new neighbors the requesting routing table command and waits for their routing table. Finally, it sends its own routing table to the initial node. Especially, the node will send its requesting packet after receive at least one requesting packet from any node as well as nodes send confirmed packet as much of packet as receiving.

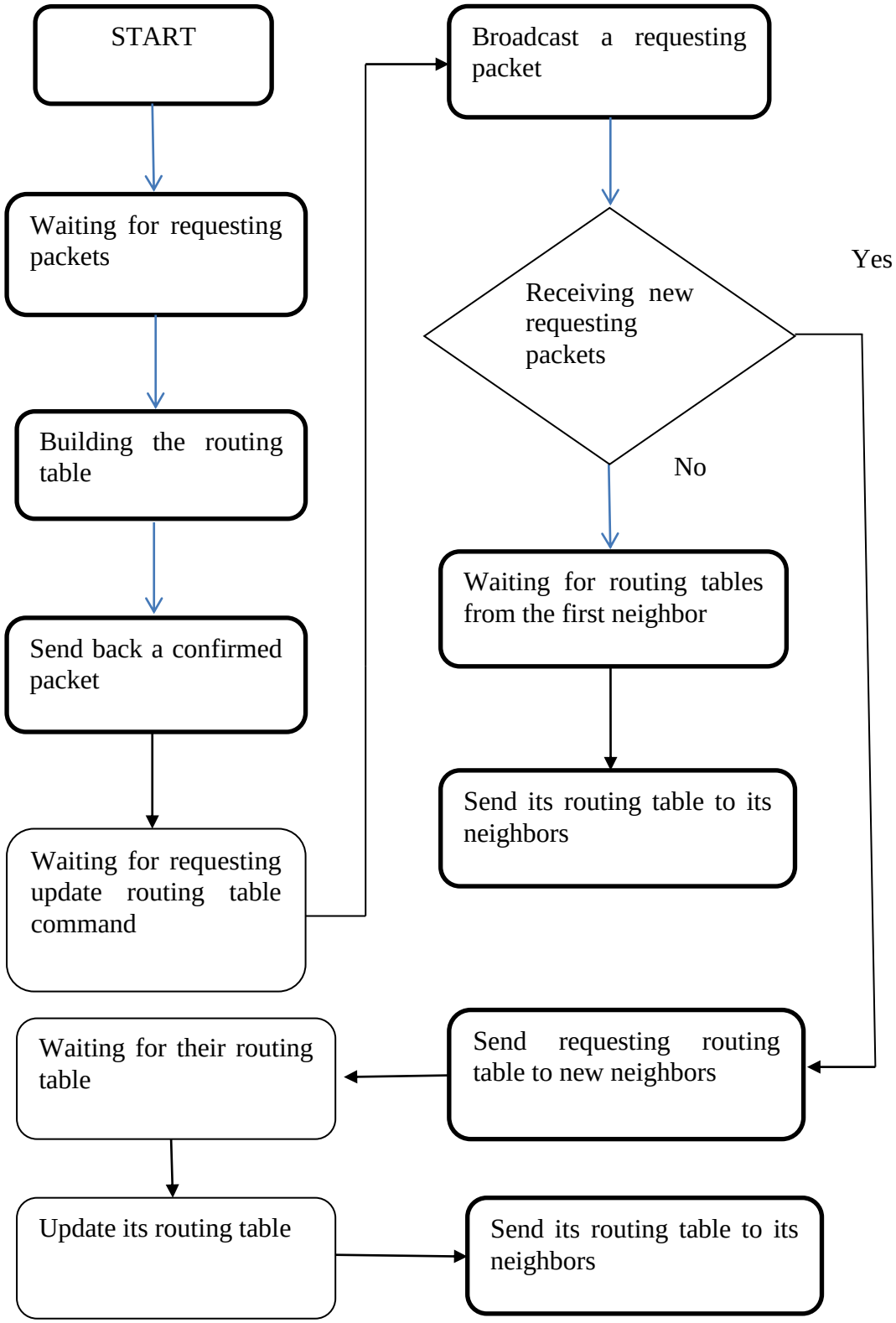


Figure 22. Flowchart of the routing protocol of Communication Node

Collision easily happens due to all nodes use the same channel to transmit. To avoid collision, Communication Nodes transmit their own requesting packet in the different time. The nodes delay in vary of amount of time which leads to difficult to calculate the timeout for RFC controller. If a node has timeout less than the delay time of a certain node, that node is not able to receive the requesting packet. Therefore, more node in the network more chance to meet collision and miss packets.

5.4.2. Collecting Data from Frequency Converter

Gateway Node

The Gateway Node acts as a gateway between the wireless sensor network and the datalogging PC. After finishing routing process, gateway has information of the network. PC sends commands to Gateway Node and then Gateway Node retains those packets. Basing on the routing table which is built before, Gateway Node finds the shortest path to reach a destination. The Gateway Node waits for response packets from other communication nodes which collect information from frequency converters.

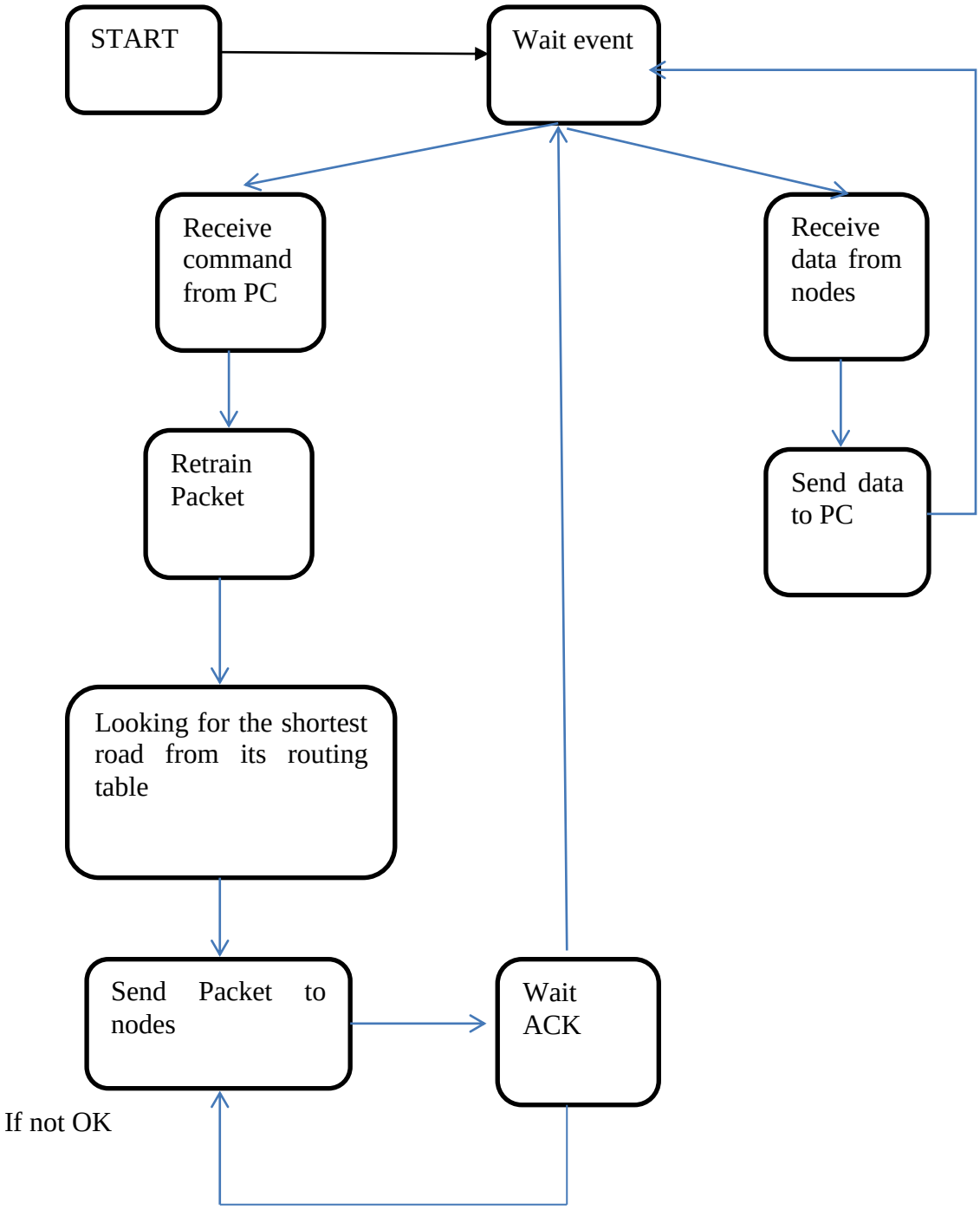


Figure 23. Flowchart of the Gateway Node operation

Communication between PC and the Gateway Node

The communication between the Gateway Node and the data logging PC is established as a virtual serial port over USB with baud rate 921600. PC sends configuration, start and stop command to Gateway Node. Packet structure for communication between Gateway Node and PC is shown in Table 19.

Table 19. Packet structure for communication between the Gateway Node and the data logging PC

Start	Length	Node Address	Type of packet	Data	END or CRC
0xAA	1 byte	2 bytes	1 byte	n bytes	0x03

Length field in each packet indicates length of data part of each message. There are 5 types of packet in this communication link.

Table 20. Packet type values for communication.

Type of Packet	
0x02	Configuration
0x03	Start
0x04	Stop
0x05	Read
0x06	Data from Nodes

PC sends configuration packet to Gateway Node

Table 21. Configuration Packet

Start	Length	Node Address	Type of packet	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 byte		1 byte	2 bytes		2 bytes	0x03

PC sends start command to Gateway Node with specify destination. There are two types of data inside this command.

Gateway Node handles the difference between continuous logging and log only changes and sends data to PC based on those settings. If PC sends 0x01 (continuous logging) wireless network starts to read data continuously. If 0x02 (log only changes), Gateway sends response packet when data got from communication nodes is different from the previous.

Table 22. Start command

Start	Length	Node Address	Type of packet	Data	END or CRC
				start command	
0xAA	0x01	0x00FF	0x03	0x01 continuous logging 0x02 log only change	0x03

PC send read command to Gateway Node

Table 23. Read command

Start	Length	Node Address	Type of Packet	End or CRC
0xAA	0x00	2 bytes	0x05	0x03

Data message from Gateway Node to the data-logging PC

Table 24. Response packet form the Gateway Node to PC

Start	Length	Node ADD	ToP	Data								END
				Read command	memory index/ID 1	Data from index 1	…	Read command	memory index/ID 10	Data from index 10	Time Stamp	
0xAA		2 bytes	0x06	1 byte	2 bytes	4 bytes		1 byte	2 bytes	4 bytes	4 bytes	0x03

Time stamp in his command is amount of read command the Gateway Node sent to Communication Nodes.

PC sends stop command to the Gateway Node to inform stopping reading packet from wireless network.

Table 25. Stop message

Start	Length	Node Add	ToP	Data	End
1 byte	1 byte	2 bytes	1 byte	0 byte	1 byte
0xAA	0x00	0x00FF	0x04	NULL	0x03

Communication between Gateway and Communication Nodes

Nodes use current implemented RFC software to communicate with each other. IEEE address of the gateway is set to 0x0000. Addresses of other nodes are from 0x0001 to 0x000n.

Packet structure of communication in wireless network

Table 26. Packet structure for wireless communication

Destination Add	Desired destination Add	ToP	Data	CRC
2 bytes	2 bytes	1 byte	n bytes	1 byte

Gateway Node received a packet from PC. That packet contains address of desired destination which PC wants to take information. Gateway Node looks for that address in its own routing table. If the routing table hasn't that address, gateway will broadcast a packet with address 0xFFFF and put desired address of PC to the packet it will send. Otherwise that address is mentioned in the Gateway Node's routing table, Gateway Node sends it based on that.

CRC checks packet from ToP byte to end of data field. CRC field is used for checking packet right or not.

Gateway Node sends configuration packet, read and response request command to Communication Nodes. Sender waits receiving node to send ACK or NACK according the message integrity. If ACK is requested, the ACK message has the same ToP value as in original received message.

Gateway Node configures Communication Nodes by sending a configuration packet. ACK requested for being sure that all Communication Nodes have received the packet. If ACK is not received, Gateway Node will resend 3 times.

Table 27. Configuration packet structure

Destination Add	Final Destination Add	Hop count	ToP	Data							CRC
2 bytes	2 bytes	1 byte	0x02								1 byte

After receiving start command from PC, GW sends read packet to all communication nodes based in time interval it got from configuration packet.

Table 28. Read Packet

Destination Add	Desired Destination Add	Hop count	ToP	Data	CRC
2 bytes	2 bytes		0x05	Sequence number 4 bytes	1 byte

Gateway Node sent read packet, it waits for 100 ms and sends response request to each node to ask response. Gateway Node will wait maximum 50 ms for response. If response packet lost, Gateway Node will resend request packet one more time to avoid losing information.

Response request from to Gateway Node

Table 29. Response Packet

Destination Add	ToP	CRC
2 bytes	0x06	1 byte

Communication Nodes

The software flowchart implemented in Communication Node operation is shown in the Figure 24. Once the system starts, Communication Node which connected with Frequency Converter, tries to communicate with Frequency Converter to make sure that connection between it and Converter is good. After having a good connection, it waits for a command from Gateway Node. There are two types of command might be sent from Gateway Node: configuration, and response request command.

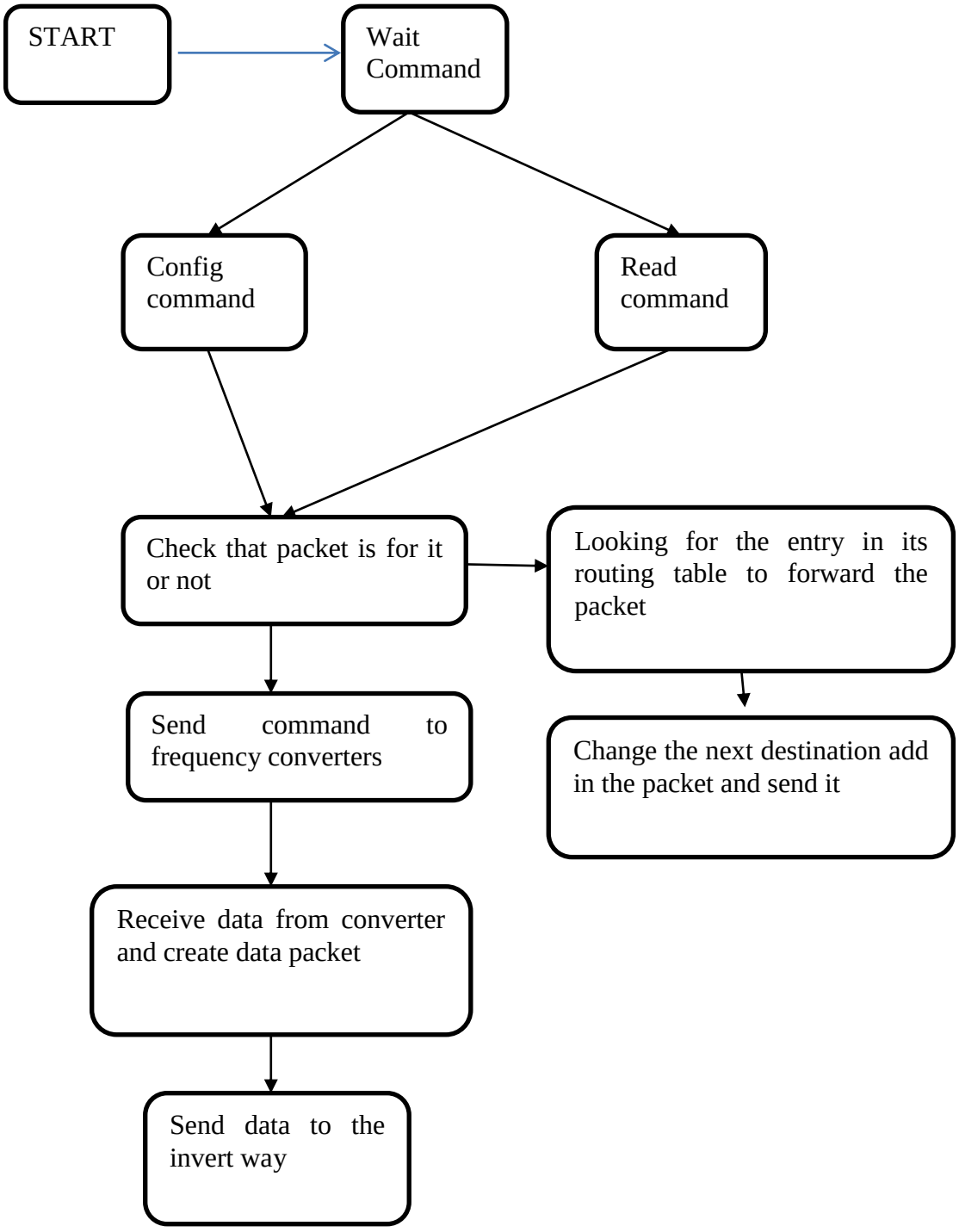


Figure 24. Flowchart of the Communication Node operation

Communication Node receives configuration command from Gateway Node and it sends back ACK message after finding the best road to get gateway from its own

routing table. Communication Node operation checks whether this configuration command already has been saved in flash memory from last command. If there is any change in configuration then Communication Node will save new configuration data into flash memory. Packet structure of configuration packet is the same in chapter 2.2.1.

Communication Node receives read command from Gateway Node then it creates a packet shown in Table 30 and sends it to frequency converter. After receiving read packet from Communication Node, frequency converter sends back data associated to that parameter. Table 31 shows the packet structure of data received from frequency converter.

Table 30. Packet structure of data sent to frequency converter

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 app always 1		DLE	ETX	
0x10	0x02	0x40	0x0B	0x00	0x00 MSB	0xBC LSB	0x00 MSB	0x01 LSB	0x10	0x03	0x66

Table 31. Packet structure of data received from frequency converter

Start chars		Command	Object	Data type	Data		stop chars		CRC
DLE	STX	Response	Variable object	int	memory location		DLE	ETX	
0x10	0x02	0xC0	0x0B	0x03	0x00 MSB	0x00 LSB	0x10	0x03	0x52

Communication Node send back to Gateway Node data packet received from Frequency Converter. The packet structure of data message sent from Communication Node to Gateway Node is given in the Table 32

Table 32. Packet structure sent from communication node to Gateway

Destination Add	CS	Data								CRC
		Sequence number	Read command identifier	memory index/ID 1	data from index 1	…	Read command identifier	memory index 10	data from index 10	
0x00xx	0x06	4 bytes	1 byte	2 bytes	4 bytes		1 byte	2 bytes	4 bytes	1 byte

For the Communication Nodes without a connection with Frequency Converter, no need to save configuration data to flash memory. If they receive response packet from Communication Nodes connected with frequency converter, they add one more field to this packet to inform Gateway Node which is the source address transmitted. The packet structure of data message is shown below

Table 33. Response packet structure of Transparent Node

Destination Add	Source Add	CS	Data								CRC
			Sequence number	Read command identifier	memory index/ID 1	data from index 1	…	Read command identifier	memory index 10	data from index 10	
0x00xx	0x00xx	0x06	4 bytes	1 byte	2 bytes	4 bytes		1 byte	2 bytes	4 bytes	1 byte

Communication node will send ACK message to gateway to inform it received successfully the configuration data from Gateway.

Table 34. ACK packet

Destination Add	CS	Data	CRC
0x00xx	0x02	0x01(ACK)	1 byte

6. EXPERIMENTS AND RESULTS

6.1. Experimental Setups

The system architecture contains of UWASA Nodes, Frequency Converters, PC. Its performance is evaluated through following experiments. All experiments go through three main cases which are three numbers of hop of the network, one, two and three hop counts.

One Hop.

One hop means Gateway Node can communicate directly with the Communication Nodes which are connected with frequency converters.

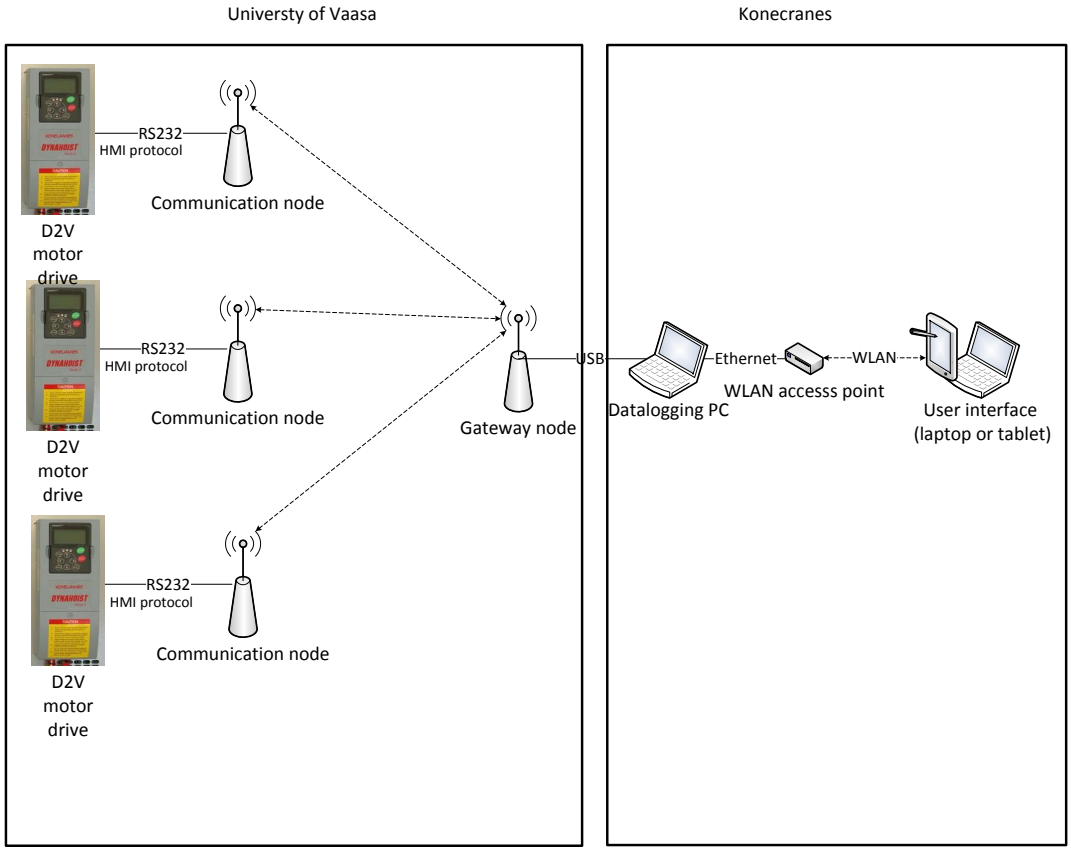


Figure 25. The One Hop System

Two hops

Two hops system means the Gateway Nodes communicates with its desired destination through a Transparent Node. A Transparent Node's task is to lead Gateway Node to its destination. It retains packets from gateway basing on its own routing table and sends the packet to destinations. It is unnecessary to store any information of frequency converters in transparent nodes.

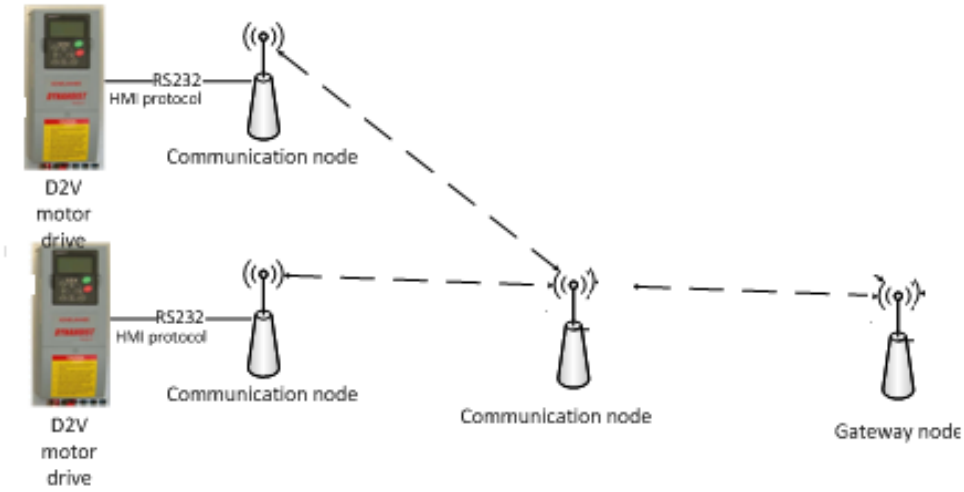


Figure 26. The Two Hops System

Three Hops

Three hops system is a kind of system which needs two transparent nodes to help carry packet from the Gateway Node to its wanted destinations. It is more complicated than one or two hops system since nodes have more neighbors and more paths to reach destinations.

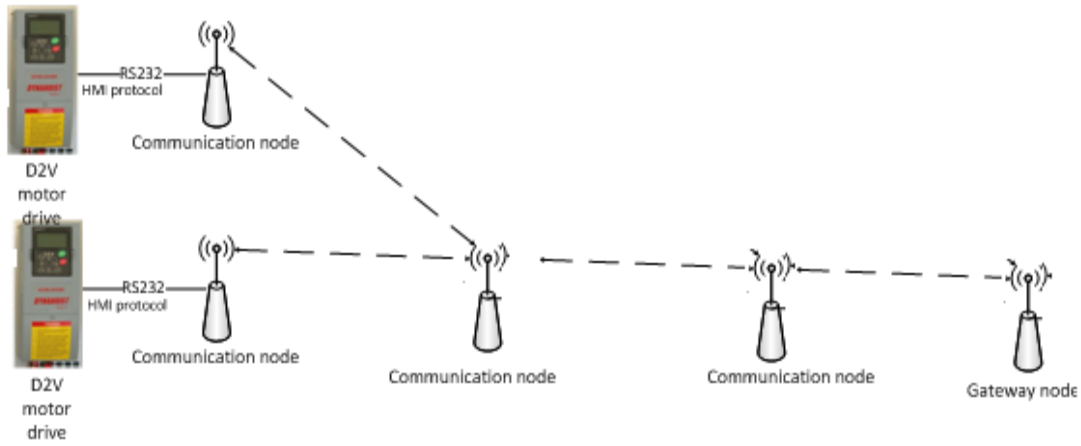


Figure 27. The Three Hops System

The first experiment is to measure how accurate of the system. Accurate level of a system is what percentage all nodes build their own routing table correctly. All three cases were considered. These cases were one, two and three hops of the network and then compared different levels between three cases. The accurate level is measured basing on the formula 1 below.

$$Accurate\ level = \frac{Number\ of\ times\ of\ success\ routing\ table}{Number\ of\ Total\ times\ building\ routing\ table} . 100\% \qquad (1)$$

In the second experiment, how the network changing when certain nodes joined or left the network. There were 7 nodes in the network and the experiment went through 2, 3 and 4 nodes respectively leaving or joining the network.

The third experiment is to measure the convergence time of the routing protocol. Convergence time means how long all nodes in the network need to completely build their own routing table. In the experiment, there are a diversity of distance as well as number of hops of the network. In the scenarios, number of hop between the sink and

communication nodes was respectively set to 1, 2, 3 hops. Also, the distance between nodes was 5, 10, 20, 50 meters. They were placed one meter above the ground level.

The fourth experiment is to measure delay time of the system. After the network converged, how long it takes since the gateway sends a packet to a destination under the routing protocol. We measured delay time in three cases which are 1, 2 and three hops in the network. The Gateway Node sends packets in which a maximum one is 90 bytes to a certain node. That node might in or not in the gateway's routing table and we could see how the Gateway Node handles those cases. Plus, since the Gateway Node received a response packet from communication nodes, it sends to PC via USB. Then we can capture the data packets from the port of PC by using a serial terminal.

RealTerm is a terminal software for capturing and controlling data streams. Here, it is used to capture and analyze the packet. One example about using the RealTerm is shown in Figure 28. The message structure contains bytes which are explained in Chapter 5.

The last experiment is to measure the packet loss as a function of the number of hop of the network as well as the distance between the UWASA Nodes. The packet loss can be calculated by formula 1.

$$Packet\ Loss = \frac{Number\ of\ Lost\ Packets}{Number\ of\ Total\ Packets}.100\% \qquad (2)$$

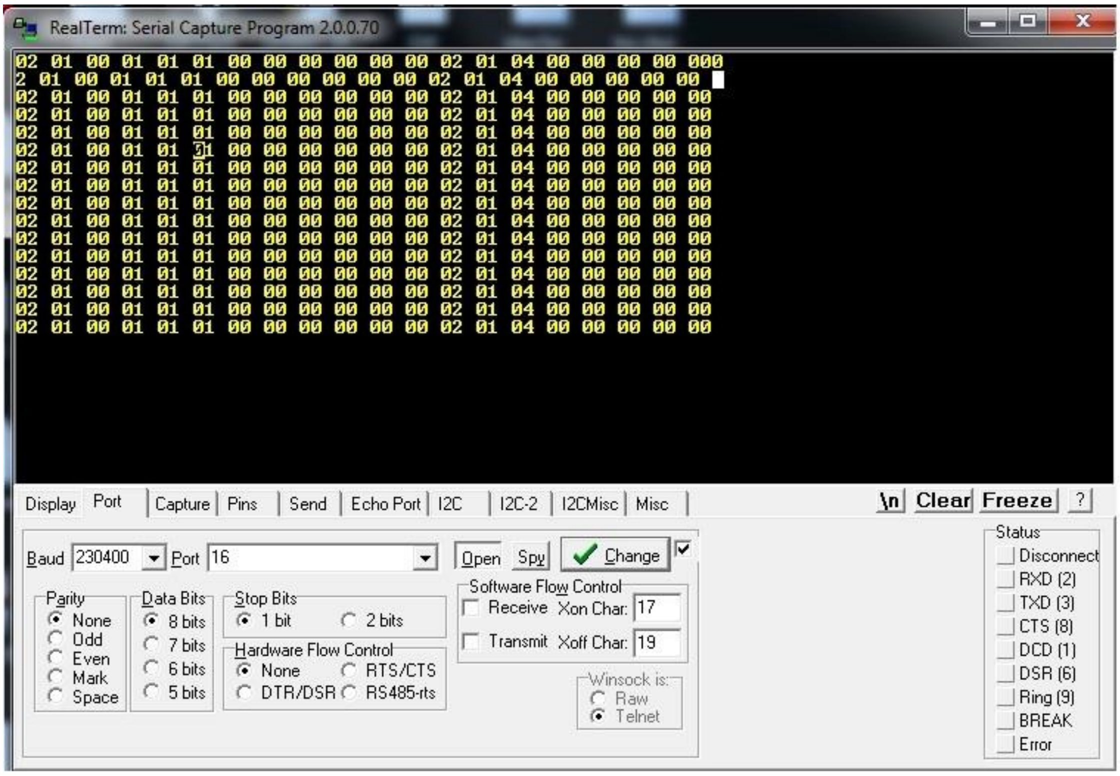


Figure 28. One test example in RealTerm, which is a serial capture program.

6.2. Results and Evaluation

6.2.1. Measuring Accuracy Levels of The System

One Hop System

In the experience, two nodes are able to communicate directly each other. Gateway Node builds its own routing table after routing process ended. The distance between two nodes is 10 meter. Figure 29 shows the gateway’s routing table.

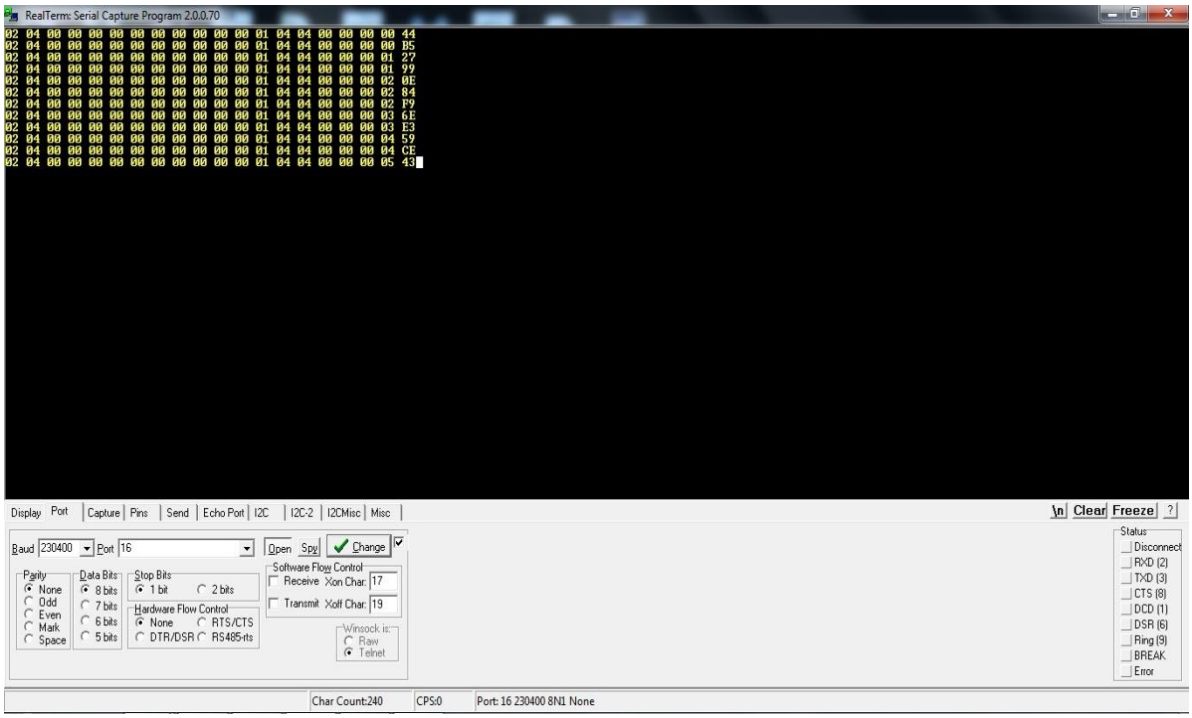


Figure 29. The routing Table of the one Hop System

The Gateway Node always builds the correct routing table after 60s. After 5 minute the system has been running, one more node joined to the network. The Gateway Node will rebuild its own routing table as Figure 30.

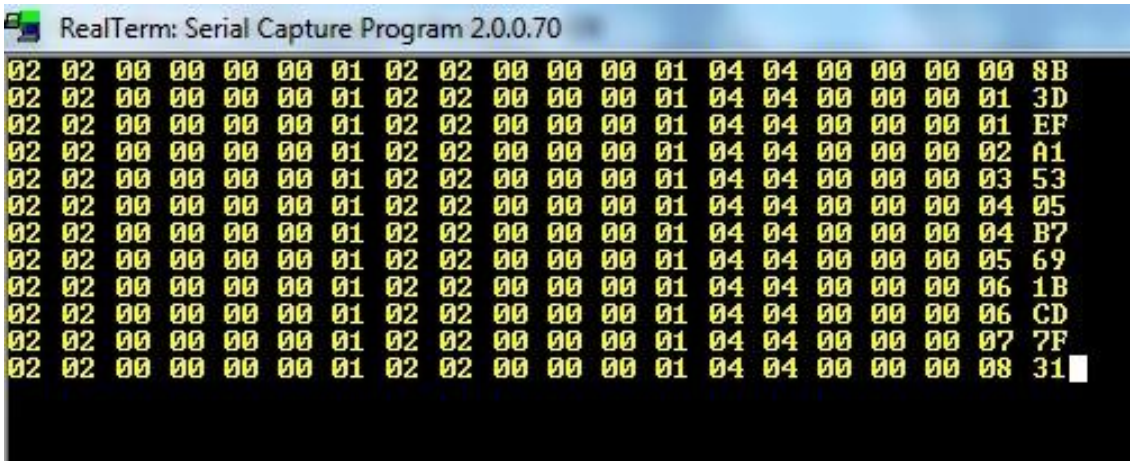


Figure 30. The routing Table of Gateway after one node joined

The group of three bytes in the routing table is used for information of one node. The first byte of the group expresses number of hop needed to reach that node, the second one is the entry to follow to get destination and the last one is address of the destination. As the routing table above, the first three byte group shows gateway can get itself by following node 4 with hop count of 1 but actually the first group is not used at all.

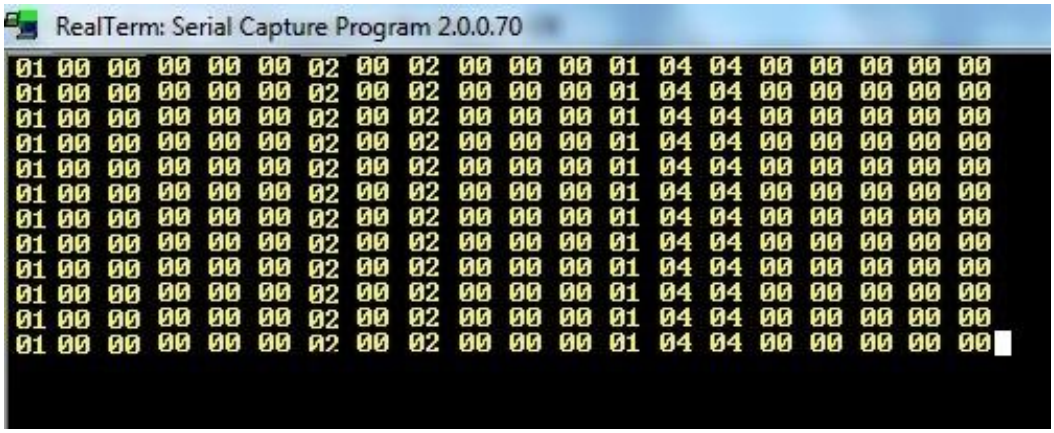


Figure 31. The routing table of Node 2

All nodes in the network build their own routing table. The Figure 31 shows the routing table of node 2 in the network which contain of three nodes can communicate directly each other.

Two Hops System

In the experience, there are three nodes in the network. Gateway Node can communicate directly with node 4 and node 4 is able connect with node 2 and node 1. Gateway builds its own routing table after routing process ended. The distance of a hop is 10 meter. Figure 32 shows the Gateway Node's routing table. Gateway Node can connect to Node 1 or Node 2 through Node 4.

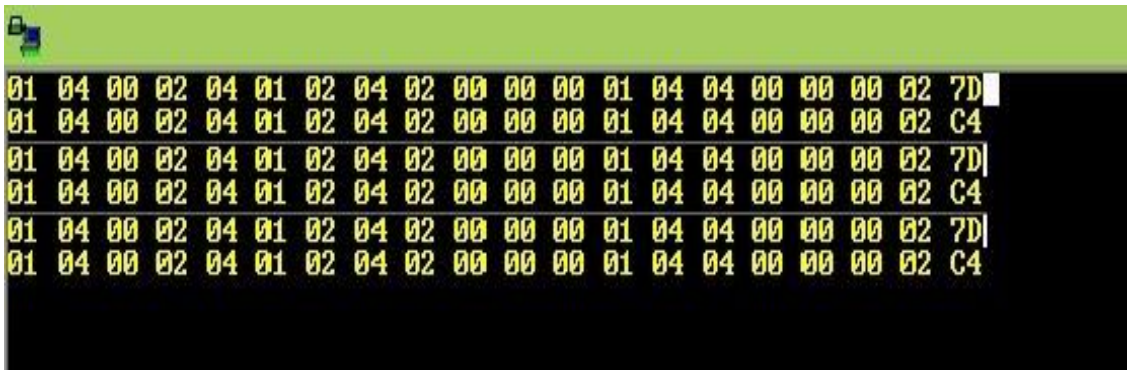


Figure 32. The routing table of the two hops system

When one node left, all nodes will rebuild their routing tables. Gateway Node's routing table is expressed in Figure 33.

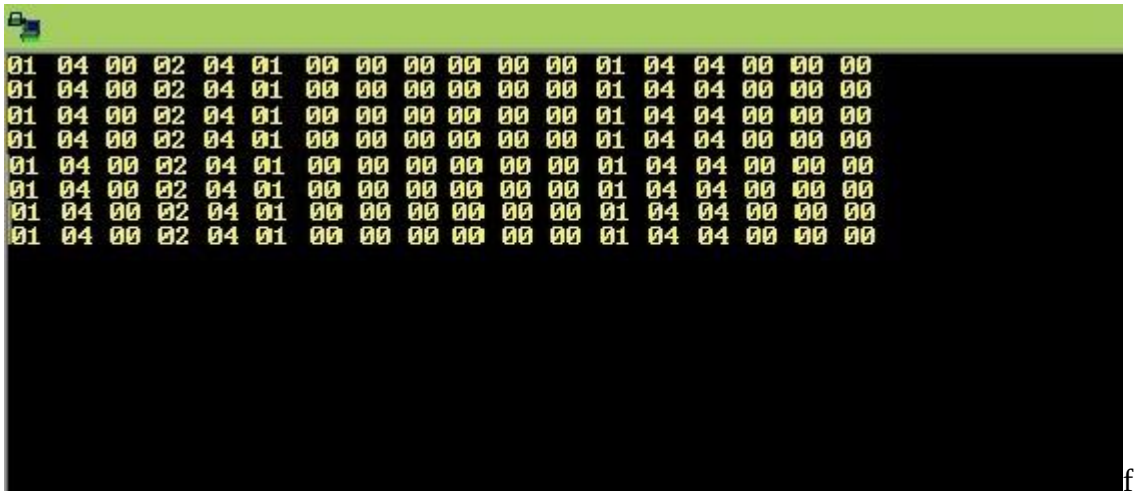


Figure 33. Routing table of gateway when one node left.

In this experiment, Node 4 can communicate with three nodes directly that means hop count is 1. Node 2, node 1 and gateway are three nodes in the routing table of node 4 as

Figure 34. For example, Node 4 can send a packet to node 1 with entry gate is node 1 and hop count of 1.

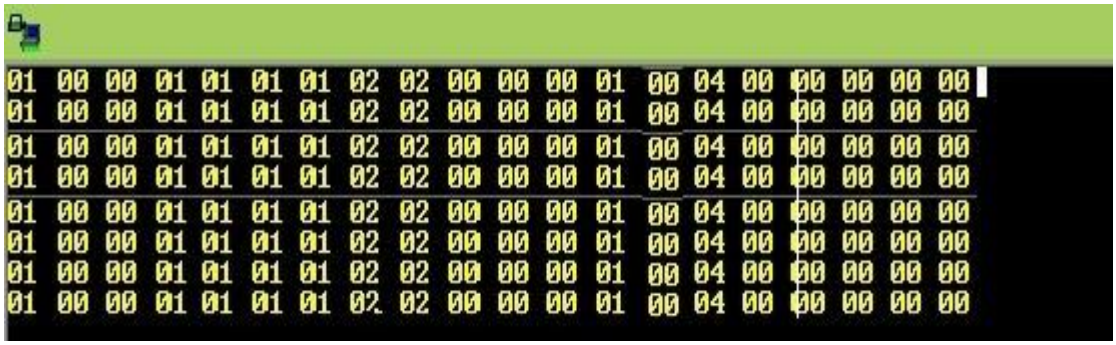


Figure 34. The routing table of Node 4

Three Hops System

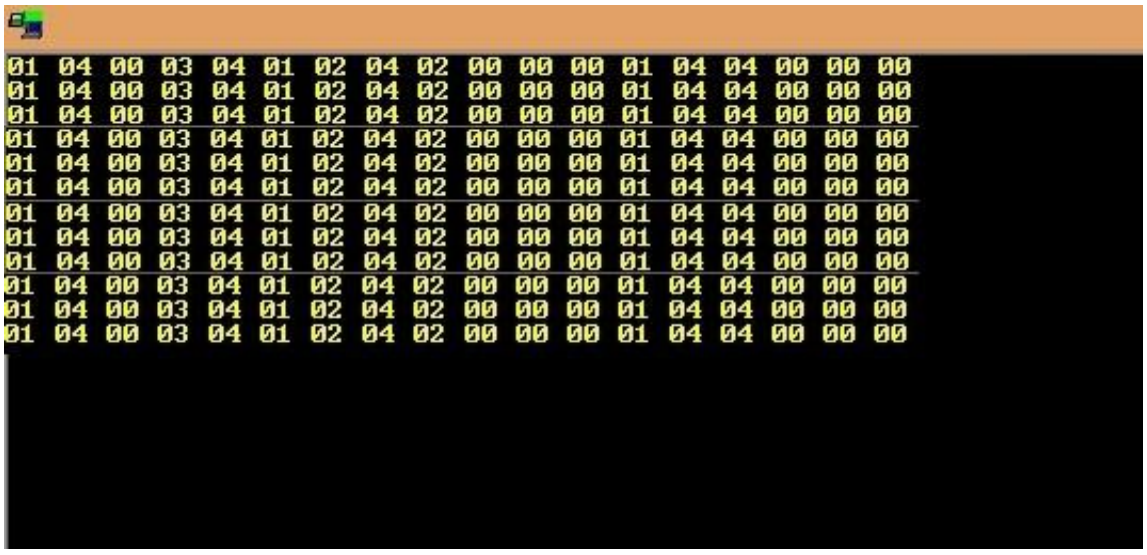


Figure 35. The routing table of the three hops system

Figure 35 illustrates the routing table of Gateway Node in the three hops system. Nodes which addressed 0x00001, 0x0002, 0x0004 and gateway 0x0000 are put in the distance of 10 meter between two nodes. After the network converged, gateway has its own routing table. Gateway Node can send directly to node 4 with hop count of 1. Gateway Node wants to send a packet to Node 1, it will send to the Node 4 firstly and after that Node 4 will check in its routing table and forward the packet to Node 1.

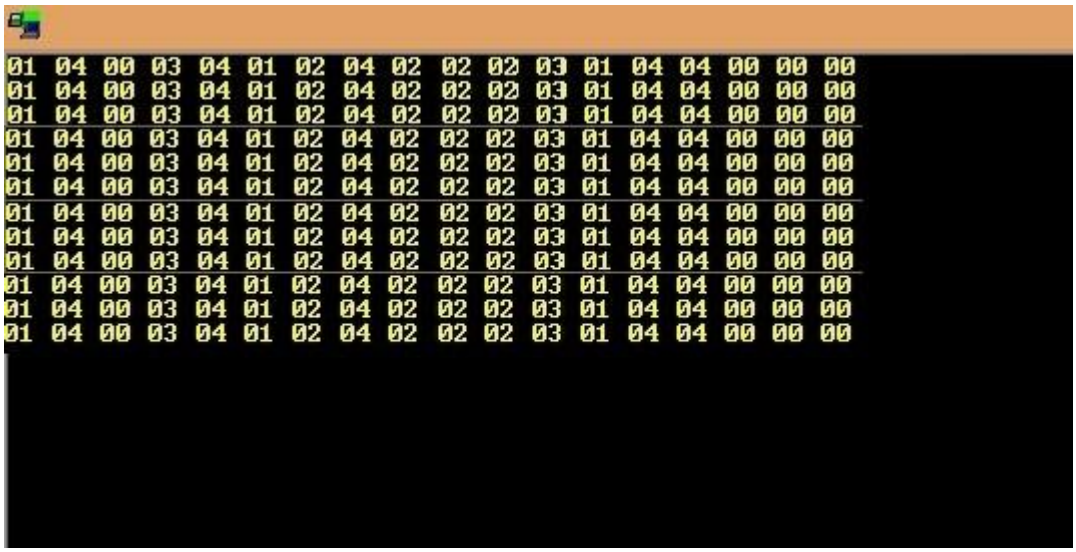


Figure 36. Routing table of Gateway Node when one node joined

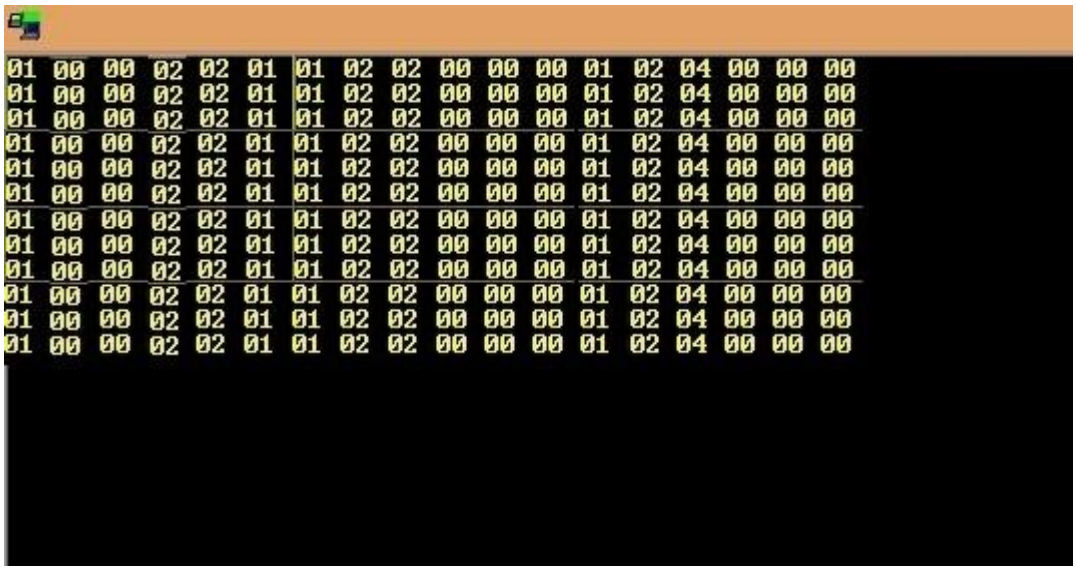


Figure 37. The routing table of Node 4

Figure 37 and 38 show the routing table of Node 4 and Node 2 in the network.

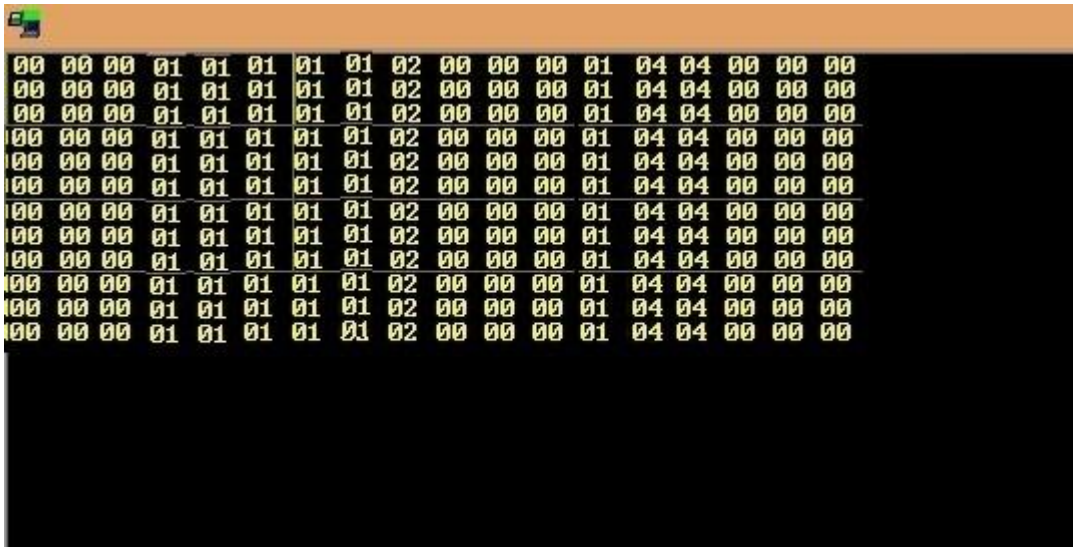


Figure 38. The routing table of node 2

When system is running with four nodes and Node 3 joins to the network, all nodes rebuild their routing tables. In the Gateway Node's routing table as figure 33, Gateway Node can get Node 3 by following Node 2 with 2 hops. It also means Node 3 can communicate directly with Node 2.

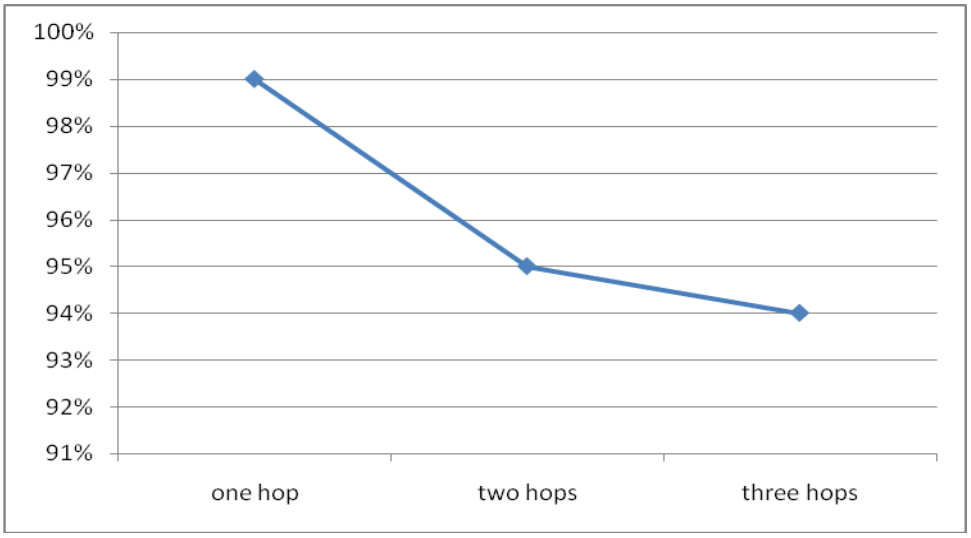


Figure 39. Accuracy percentage of routing protocol

Figure 39 shows accuracy percentage of the routing protocol. This measurement was executed with 200 times for three cases one, two and three hops system respectively. With one hop system, the Gateway Node just missed one node among 5 nodes two times under 200 times of routing. For two hops system, there were 5 times Gateway Node's routing table couldn't get information of one node from the node it connected directly with. It is more complicated in the three or more hops system. The number of losing node is much more but the routing protocol is able to find another way to get destinations which is obviously not the shortest path.

6.2.2. Time Convergence of The Routing Protocol

Time convergence of the network means how long do nodes need to complete their routing tables. Obviously, if the network has more hop counts, time convergence is

much more. In this measurement, after 100 times of routing process, mean of those and variance of the process were calculated.

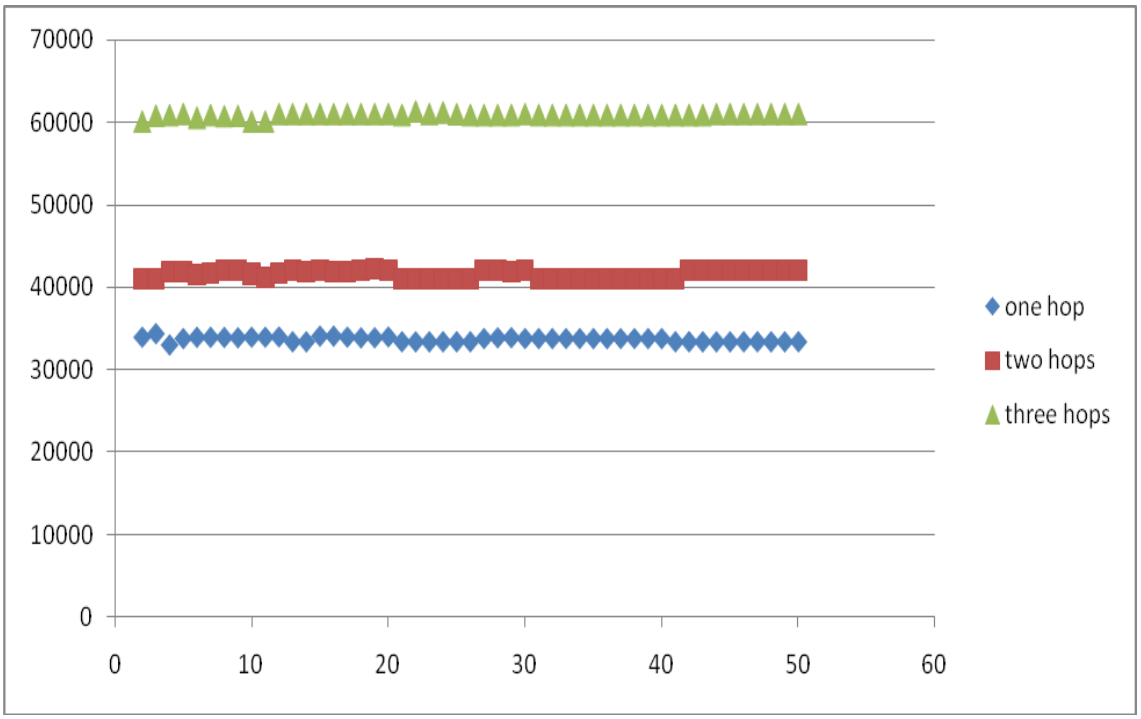


Figure 40. Time convergence of the routing protocol.

Figure 40 shows time convergence of three systems. The x axis is how many times the measurement was executed. The y axis is about time in ms. For one hop system, after 50 times transmission, mean of time convergence is 31000 ms with variance of 100 ms and the system is stable. With the two hops system, mean of time convergence is 41000 ms and variance is 120 ms. For the three hops system, mean of time convergence is 60100 ms and variance is 150 ms. It is obvious more hops of the system much more time the system needs to converge. Nevertheless, variance is increased from 100ms to 150 ms even more in more hops system. With the three hops system, 150 ms is acceptable.

6.2.3. Time Delay of Transmission under The Routing Protocol

In this experiment, the Gateway Node sends one packet to a certain node after routing process finished. Time delay of transmission means how long it takes since the Gateway Node sends a packet to gateway receive a response packet.

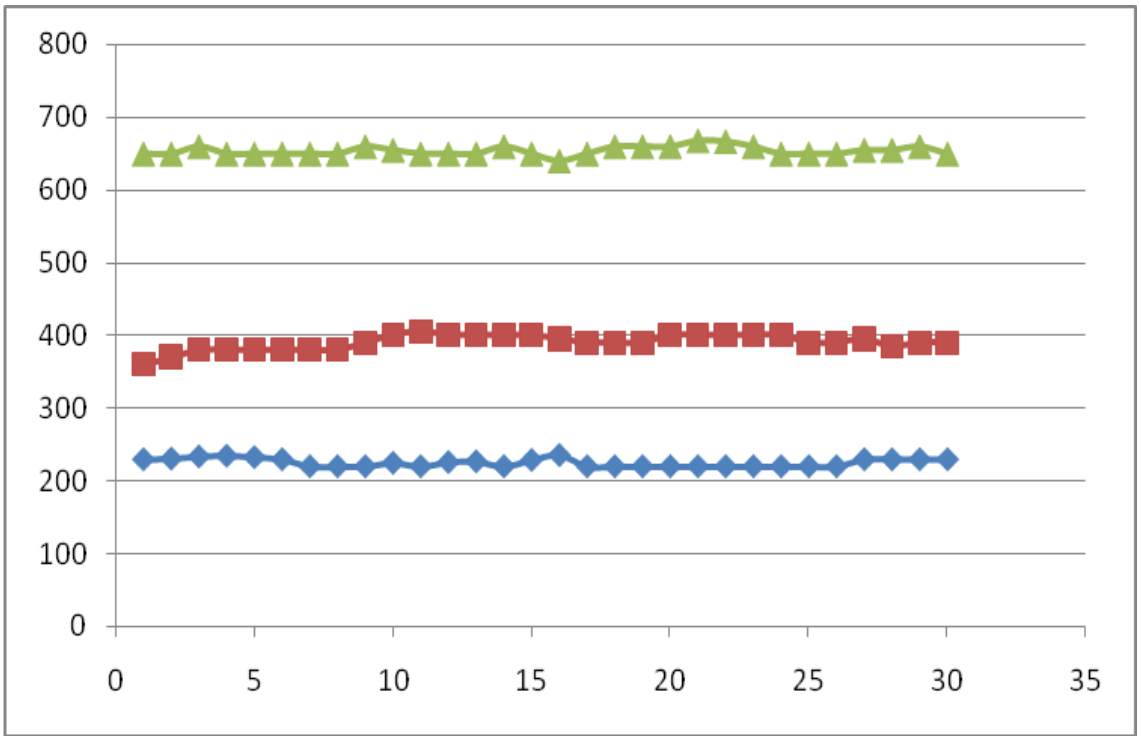


Figure 41. Time delay of the three systems

Figure 41 shows time delay of the systems. With the one hop system, the average time delay of communication is around 230 ms and variance is 10 ms. The average time delay in the two hops system is about 400 ms with variance of 20 ms. The last system-three hops system has average time delay of 650 ms and a variance of 20 ms.

6.2.4. Packet Loss

In this experiment, the Gateway Node collects information from frequency converters under the routing protocol. The Gateway Node will send three kinds of command to its desired destination which is mentioned in Chapter 5.

Packet loss of three systems after sending 200 request commands in every 500 ms for each command was compared. With the one hop system, packet loss is about 0,3% and the system is stable. With the two hops system, packet lost is around 2% even sometimes one node gets stuck and packet follows another path to get its destination. For the three hops system, packet loss is 3% and the routing tables are changed sometimes.

The Figure 42 shows the response packets the Gateway Node received from Nodes 1 and Node 6 which are connected with Frequency Converters. The Communication Nodes connected with frequency Converters gather 10 parameters from frequency converters and send back to Gateway Node. The structure of all commands is described in detail in Chapter 5.

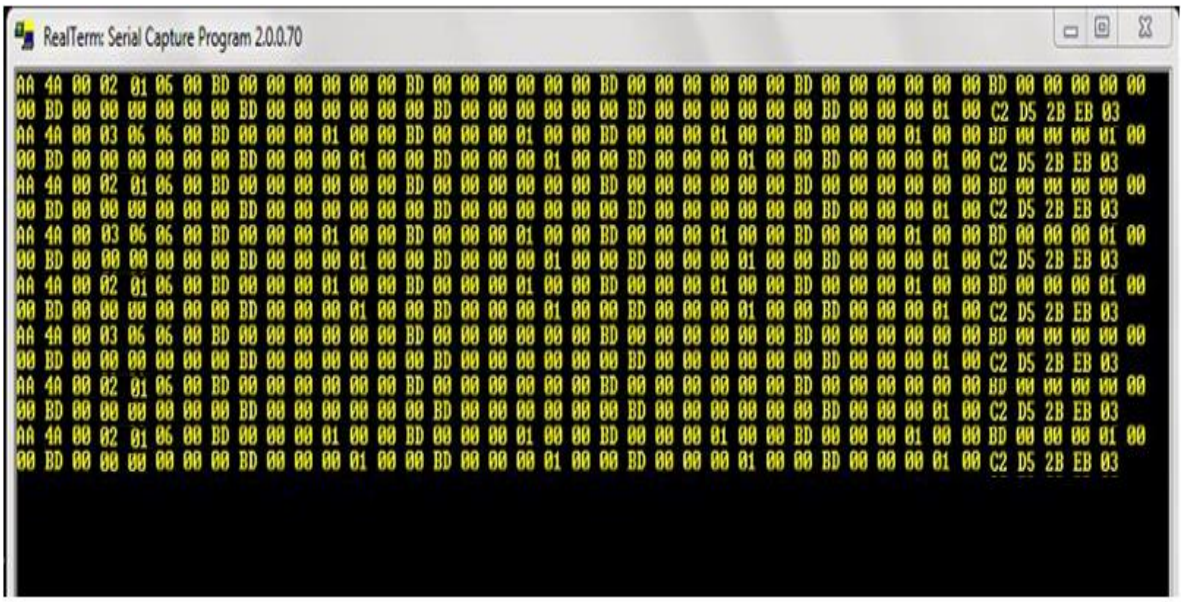


Figure 42. Response packet the gateway received

7. CONCLUSIONS AND FUTURE WORK

7.1. Conclusions

The performance of the developed wireless sensor system has been through the experiments discussed in Chapter 6. It can be used to collect data from the frequency converters as well as to control frequency converters. This work is completed by developing a routing protocol and a logging data protocol. Under the routing protocol, the nodes collect data from the frequency converters over a multi-hop wireless sensor network. Then the data is transmitted through the Gateway Node to the data logging PC.

Generally speaking, this study has achieved two main goals. The primary goal is to collect information from frequency converters from remote positions by using a wireless communication protocol. The first goal has been fixed. The limitation of distance of the one hop protocol is fixed. Distance of communication between gateway and frequency converters is increased up to 4 hops in which each hop distance is about 10 meters. Moreover, the system is flexible, whenever some nodes left or joined the network and transmitter is able to get its desired destination in the best path.

The second goal is to gather data from frequency converters. Under the routing protocol, nodes in the network exchange information which got from frequency converter. Gateway Node collects all information and send to PC where user can read by Realterm software.

Consequently, this protocol can be developed for various industries. The system is used to collect data from remote devices as well as control those devices by using an application designed on PC or mobile phone.

7.2. Future Work

Firstly, there is a need for further study to improve the stability of the system when the system is not stable. When some nodes move, or there exists more disturbances in the radio environment, or many undesired signal, they impact much on the system reliability. This protocol ignored all effects of these on the system and tried its best to build an accurate routing table. Especially, when some nodes are moving, the routing table should be changed fast enough and be more accurate for the network. Practically, the protocol cannot be updated fast enough as well as accurate in this case. The routing table might give a wrong path to get destination and create an unwanted loop in the network. We might need to research a algorithm to deal with mobility.

Secondly, to improve the reliability of the wireless communication, the antenna of the node can be developed further so that the transmitting signal gain can be increased. Thus the reliability of communication system can be increased.

Thirdly, the data security of the wireless communication can be further improved if it is required. In some industrial applications, the data security is more critical, so that the encryption and decryption must be enabled.

Finally, the most important one is metric can be developed more accurate for building a routing table. Most often in the literature, the routing tables are built by using a hop count metric. That is not the best tool to build a routing table. For example, after a node built its own routing table, and it sends packet through a shortest path to reach its destination but status of that path is too bad. That means there is a lot of packets on that path already or its bandwidth is so small comparing with another path to get the same destination. Thus, according to that issue, we need to measure metric on some parameters like bandwidth, load on the link, time delay. That would be more accurate to build routing tables.

REFERENCES

Akyildiz, Ian F., Weilian Su, Yogesh Sankarasubramaniam & Erdal Cayirci (2002). *A survey in Sensor Networks*. IEEE Communication Magazine, Volumn: 40, Issue: 8, Aug.2002.

Azzedine Boukerche,PhD. *Algorithms and Protocols For Wireless Sensor Networks.*Page(s) 133- 159. Ottawa, Canada, University of Ottawa ,2009.

Barry, Richard (2009). *Using the FreeRTOS Real Time Kernel: A Practical Guide* [online]. Available from the Internet: <URL: ftp://ftp.cs.Sjtu.edu.cn:990/hongzi/embedded%20systems/reference%20books/Using+the+FreeRTOS+Real+Time+Kernel+-+a+Practical+guide.pdf>.

Callaway E.H., Jr. (2003). *Wireless Sensor Network: Architectures and Protocols*. CRC Press LLC, Florida, USA, 324p.

Cisco Networking Academy (2014). *Cisco Networking Academy's Introduction to Routing Dynamically*. Available from Internet: <URL:http://www.ciscopress.com/articles/article.asp?p=2180210&seqNum=7>.

Cuhac Caner, Huseyin Yigitler (2012). *The UWASA Node Reference Manual 3.0.0*. Vaasa, Finland: University of Vaasa, 2012.

Montenegro G., Kushalnagar N., *Transmission of IPv6 Packets over IEEE 802.15.4 Networks*, Internet-Draft, IETF, September 2007

N. Salman, I. Rasool, A.H.Kemp. *Overview of the IEEE 802.15.4 standards family for Low Rate Wireless Personal Area Networks*. Wireless Communication System (ISWCS), 2010 7th International Sysposium on 19-22 Sept. 2010, Page(s)701-705

Pahlavan K. & Krishnamurthy P. (2002). *Principles of Wireless Network*. Prentice Hall PTR, Upper Saddle River, USA, 581p.

Shelby Z. (2008). *OEM IP-based Wireless Embedded and Sensor Networks*. The WiFi of the Embedded World. [Online]. Available: http://www.sensinode.com/pdfs/sensinode-wp1-why_ip_wsn.pdf

Shuang-Hua Yang. *Wireless Sensor Networks Principles, Design and Application*. Loughborough,UK, Department of Computer Science, University of Loughborough, 2012

Texas Instruments (2009). CC2431 Data Sheet [online] [cited 8 Jun. 2013]. Available from Internet: <URL: http://www.ti.com/lit/ds/symlink/cc2431.pdf>.

Virrankoski Reino (2012). Generic Sensor Network Architecture for Wireless Automation (GENSEN) [online]. Vaasa, Finland: University of Vaasa, 2012. Available from Internet: <URL: http://www.uva.fi/materiaali/pdf/isbn_978-952-476-387-5.pdf>.

Wikipedia. *Distance-vector Routing Protocol* [online] [cited 29 Sep. 2014]. Available from Internet: <URL: http://en.wikipedia.org/wiki/Distance-vector_routing_protocol>.

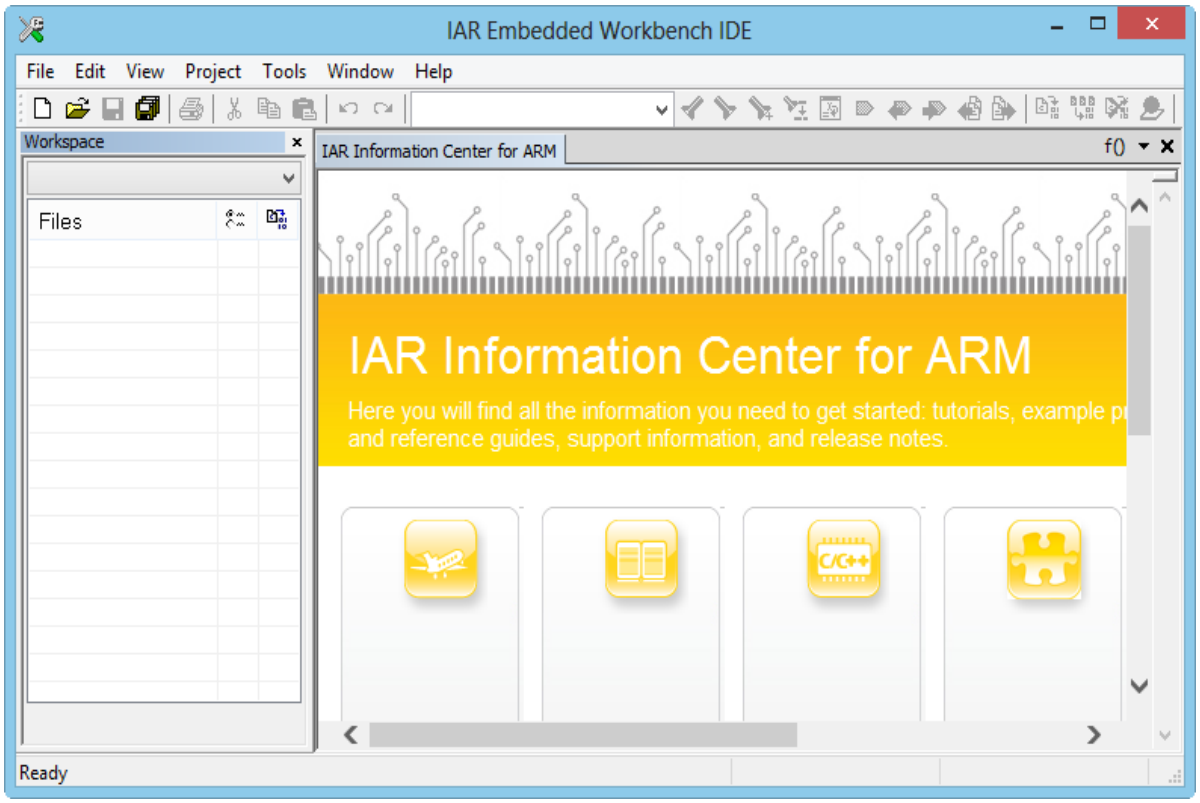
Wikipedia. Wireless Sensor Network [online] [cited 19 Sep 2013]. Available from Internet: <URL: http://en.wikipedia.org/wiki/wireless_sensor_network>.

Yigitler, Huseyin (2010). *The UWASA Node Reference Manual*. 1st Ed. Vaasa, Finland: University of Vaasa, 2011

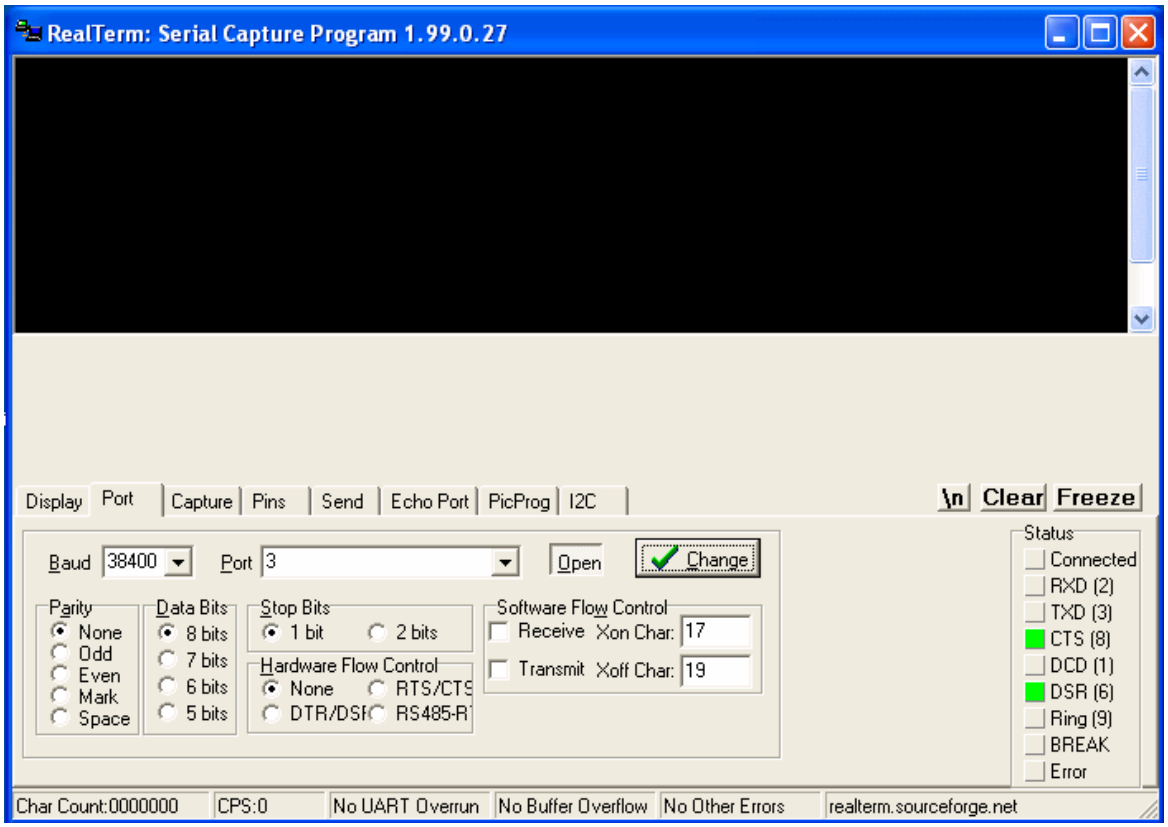
Yigitler. Huseyin, Reino Virrankoski & Mohammed Elmusrati (2010). *Stackable Wireless Sensor and Actuator Network Platform for Wireless Automation*: The UWASA Node, In: Aalto University Wireless System Group. Helsinki: Aalto University.

APPENDICES

APPENDIX 1. IAR embedded workbench for ARM IDE.



APPENDIX 2. Realterm Software



APPENDIX 3. Texas Instrument SmashRF Flash Programmer