NTNU – Trondheim
Norwegian University of
Science and Technology

Analysis of Key Industrial WSN MAC Protocols

Kenneth Johannesen Koh

Master of Science in Communication Technology
Submission date: September 2014
Supervisor: Yuming Jiang, ITEM
Co-supervisor: Hoon Oh, University of Ulsan

Norwegian University of Science and Technology
Department of Telematics

Abstract

This paper looks at two MAC protocols for wireless sensor networks for use in industrial applications developed at the Ubicom Lab, at the University of Ulsan. A theoretical comparison of the MAC protocols are performed to understand more about the benefits and downsides to both of them, and experimental scenarios to validate the theoretical analysis are suggested. Theoretical analysis suggests that BigMAC has an advantage in environments with high interference and frequent link breaks, while I-MAC has an advantage when the network topology is stable.

Sammendrag

Prosjektet ser på to MAC protokoller for trådløse sensor nettverk. Begge er utviklet på Ubicom lab, på Universitet i Ulsan. En teoretisk sammenlikning mellom protkollene har blitt gjennomført for å få kjennskap til fordeler og ulemper ved begge protokollene. Det er også blitt gjort forslag til eksperimenter som kan bli gjort for å verifisere den teoretiske sammenlikningen. Sammenlikningen tilsier at BigMAC har en fordel når topologien er veldig dynamisk, mens I-MAC har fordel når topologien er stabil.

Contents

List of Figures

Chapter 1 Introduction

1.1 Background

The adaptation of wireless sensor networks (WSN) over its wired counterpart has been a slow process. The reason for this has been the technical constraints when dealing with high interference environments and the psychological trust wired devices provide over wireless ones. The use of WSN has been gaining momentum in the last several years, and as such many industrial requirements to WSN have been set in order to fulfill safety and security. Organizations like the HART foundation[1] have started standardization of protocols to provide WSN that work reliably in high interference environments. This has helped with the trust and confidence needed to migrate from wired to wireless and has made the research of WSN protocols an attractive area, especially for automated industrial process control.

1.2 Motivation

Context-aware safety monitoring and control applications that collect data from the target environment using WSN, analyze the collected data, and judge the situation of the environment are in a growing need in industrial fields.

There are two WSN MAC protocols developed at the Ubiquitous Computing Lab, in the University of Ulsan for use in context-aware safety monitoring and control applications. These goal of these sensor networks are to collect and analyze data which are used to monitoring and alert purposes. These MAC protocols are I-MAC [2] and BigMAC [3], and they both serve to function in similar environments. Since both protocols operate in the industrial environment finding out if and how they satisfy the industrial requirements is of high interest. In order to do this however, defining what they key industrial requirements are must be done. Another interesting question is how the different implementation of these protocols may give them different advantages in certain scenarios. I-MAC uses a pure TDMA approach while BigMAC uses a hybrid scheme of TDMA and CSMA.

WiHART, although a multi-layered protocol, is an already accepted standard, and can be used as a benchmark for realization of the industry requirements. Thus, using WiHART as a light comparison to contrast how I-MAC and BigMAC realize the requirements.

1.3 Scope

This paper will analyse two MAC protocols I-MAC and BigMAC too see if if and how they satisfy industrial requirements. WiHART will be used as a light comparison as it is an already accepted standard. Parameters for the industry requirements will be defined in its own chapter. The paper will also look at the differences between I-MAC and BigMAC and how their different implementations may give advantages and disadvantages in certain scenarios, and a experimental scenarios for real test-bed will be suggested.

The original plan of the paper was to perform a theoretical analysis of the difference of I-MAC and BigMAC and suggest scenarios where one might outperform the other. The second step was to implement the protocols unto a test-bed of real TelosB nodes and perform experiments. However, due to limited time and delays in implementing BigMAC, the scope of the project will change to simply do the theoretical analysis and suggest experimental scenarios but leave the experiment itself as further work.

1.4 Report Structure

The next chapter will introduce WSN in detail and explain key concepts in WSN and some history to how the MAC protocols have evolved in order to meet industry requirements. The three following chapters after that will introduce the WiHART and the two protocols I-MAC and BigMAC which will detail the key concept of the protocols and a detailed overview of how its mechanics work. Chapter 6 will be dedicated to theoretical analysis and comparison of the two protocols. Following this will be a discussion and further work followed by the conclusion.

1.5 Methodology

An understanding of the WirelessHART, I-MAC protocols were obtained by literature study.

However, BigMAC is a protocol still under development and only an internal draft of the paper exists. A combination of access to this draft and access to some simulation results.

Chapter 2

Wireless Sensor Networks and Industrial Process Control

2.1 Introduction

Wireless Sensor Network is a form of ad-hoc network. It is characterized as having nodes spatially distributed that work as autonomous sensors monitoring the environment. It contains many of the same characteristics as that of an ad-hoc network such as battery lifetime and mobility. Some of the characteristics [4] more related to WSN is given in the list below.

- **Computing Capabilities:**
  In order to limit the cost, space and power consumption of the nodes, the computing capabilities are quite limited.
- **Power Consumption:**
  Nodes may be difficult to reach and it can be very inefficient to replace. It is therefore a neccessary that the nodes have an appropriate lifetime fit for the application it is used for.
- **Communication Capabilities:**
  Highly affected by interference, and due to node density because of sensor limitations, limits the transmission range.
- **Ability to Withstand Harsh Environments:**
  Ability to withstand harsh environmental conditions and Must be adaptable to interference and the dynamic change of network topology due to interference or death of nodes.
- **Self-organization:**
  No pre-installed network needed.
- **Multi-hop communication:**
  Nodes use multi-hop to communicate with its neighbors until reaching a sink.

- **Application Relevance:**
  WSN differ depending on what kind of application it is used for.

The network is often distributed in the same manner as an ad-hoc network with a dynamic topology, however the key difference is that the data from the sensors are relayed through the network using multi-hop until it reaches a sink station, see figure 2.1.

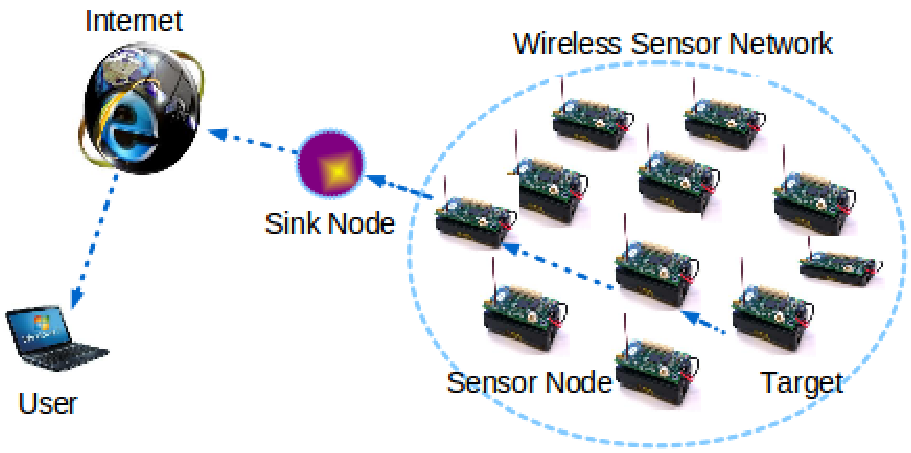


Figure 2.1: Illustration of Wireless Sensor Network.[5]

The main benefit of moving from wired to wireless is the low installation cost and high network adaptability. However, unlike its wired counterpart, wireless communication is prone to signal interference, fading and blocking. For these reasons, migrating to from wired to wireless has been a slow process, especially in application areas where reliability and timely delivery of data is of high concern.

2.2 Hardware and OS

WSN are compromised of nodes. These nodes are usually equipped with a wireless radio transmitter and receiver, a micro controller and the required electrical circuits, a sensor of some kind, and a battery. Different kind of nodes have different hardware requirements for different purposes. Microprocessor speed, memory and battery-lifetime all vary depending on the price and utility of the nodes.

The protocols to be analyzed in this project were implemented using TelosB sensor motes[6], more specifically in this case the MTM-CM3000 rev.01 model, see figure 2.2. TelosB consists of a radio transceiver, transmitter and 3 LED lights, a microprocessor and a battery pack. Sensor can be added to the motes separately.

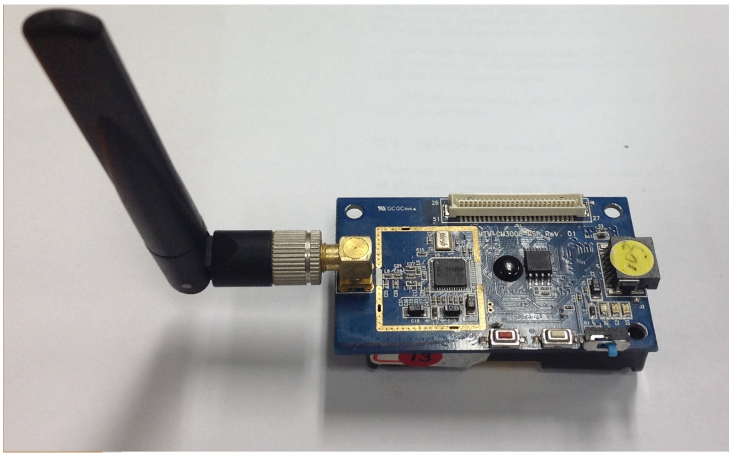


Figure 2.2: MTM-CM3000 TelosB Mote.

TinyOS [7] is an open source embedded OS targeting wireless sensor networks. It is written in nesC and is the OS used in the TelosB motes.

2.3 WSN topology

When moving from wired technology to wireless technology it is common to apply the techniques and mechanisms that has been learned through traditional network technology, modify it and apply it to wireless technology.

WSN utilize traditional topology from wired networks and ad-hock networks and modified them to work in the context of a WSN. This section will briefly introduce some common topological structures.

2.3.1 Peer to Peer

Each node can communicate directly without any centralized control. Each node can be either client or server, see figure 2.3.

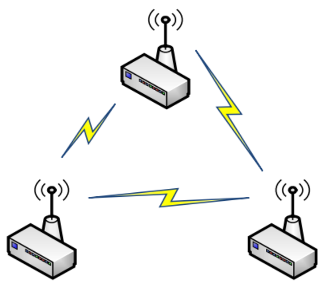


Figure 2.3: Peer-to-Peer Topology.[5]

2.3.2 Star

In this topology, the nodes are connected to a centralized hub. All communication must pass through this hub, and unlike the peer to peer model, the nodes only acts as clients while the hub acts as a server. See figure 2.4.

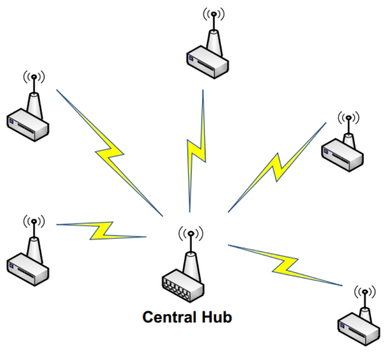


Figure 2.4: Star Topology.[5]

2.3.3 Mesh

Allows network data to hop from node to node. Each node can communicate with one another since data is sent through many nodes until it reaches its destination. These types of networks can end up being very complex. See figure 2.5.

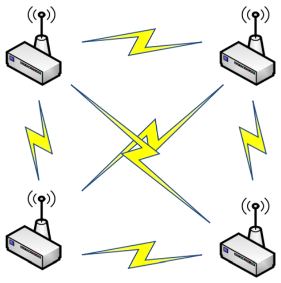


Figure 2.5: Mesh Topology.[5]

2.3.4 Tree

A tree network can be considered a hybrid of Peer to Peer and Star networks. The Root of the tree network is a a central hub which connects to other central hubs connecting several nodes to a star network in the lower levels. Both the protocols I-MAC and BigMAC utilize a tree structure. See figure 2.6.

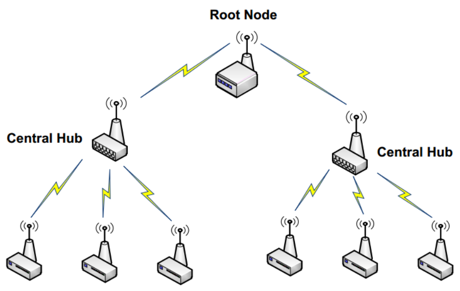


Figure 2.6: Tree-structure Topology. [5]

2.4 Application Area

WSN have many areas of application including forest fire detection, air pollution monitoring, health care monitoring and such. This paper will focus on WSN for use in industrial processing monitoring and control, and its environment. The environment in industrial processing are known to have high interference,and because of this the use of wired sensor networks have been preferred over Wireless communication. With the introduction of industrial WSN MAC protocols and protocol suits, the feasibility of using WSN for industrial application has grown.

2.5 Challenges in WSN

This section will summarize some of the most common challenges that WSN presents. The various protocols introduced later in the paper will work to solve or mitigate these issues.

In WSN reliability, memory constraints and message delays are of very high importance due to the different range and requirements of industrial application scenarios. This need for quality of service (QoS) is not easily implemented due to the limiting factors of WSN such as the computational capabilities, the memory constraints and the conservation of battery. These problems limit the adoption of the technology, but industrial oriented protocols such as WiHART, I-MAC and BIG-MAC are working on providing these QoS despite the limiting factors of the technology. These protocols will be the topic of chapter 4, 5 and 6 respectively.

Low sensor range results in dense networks, thus MAC protocols are needed to avoid collision and other problems related to wireless communication. The primary objective is to conserve power. Collision, overhearing, excess control packets, idle listening and over emitting all cause energy waste.

2.5.1 Routing and Topology

Every node in a WSN wants their sensor data to end up in the sink node. If a link dies, the route has to dynamically change and adapt in order for the package that used to travel through the dead link to still reach the sink node. This requires some mechanism to deal with dead links. A network must also be adaptable to new nodes joining the network.

2.5.2 Delay

If the delay in the WSN is too large, perhaps an important signaling arrives too late for it to be of any use, or in the case of forest fire detection, if the delay is too long, perhaps the node trying to send the signal will burn down before getting its message across.

2.5.3 Battery Lifetime

WSN may deploy hundreds of nodes over a large area. Doing battery maintenance is a costly affair when the number of nodes are high. Therefore one of the biggest concerns of a WSN protocol is to extend the lifetime of a battery by conserving power when necessary.

2.5.4 Interference

WSN are often deployed in environments where interference can be a major factor. Weather and movement can cause blocking or interference in the signal.

2.6 WSN in Industrial Process Control

In regular WSN the main concern is power consumption. However, in the industrial application the focus is also on reliability and timely delivery of data. The scale-ability and adaptability of the MAC protocol is also important due to the dynamic nature of the environment. This is especially important for safety issues and is difficult to achieve in wireless communication.

This project will look at WSN MAC protocols designed for industrial context and safety applications. WSN has many application areas and depending on the field where the nodes will be placed there will arise different limitations and requirements. This section will introduce the industrial process control environment and give a brief overview over the history of MAC protocols in this area. This section will also attempt to define some parameters that can be used to evaluate a MAC protocol with this field in mind.

2.6.1 Industrial Process Control and Context-aware Safety Monitoring

The goal of WSN in industrial process control is for it to act as a context-aware safety monitoring system and control application by using sensors. The usage of WSN has grown tremendously these recent years however, for context-aware safety monitoring systems the trend has been more slow, due to innate faults in WSN and the wireless medium itself. These faults can be counter-acted by using protocols. However, for context-aware safety monitoring systems, such protocols have not met the standard required. The environement of industrial process control are highly dynamic, meaning links break often. It also has strict time requirements for the sensor data to reach the sink and has to be reliable. The constraints of the environment combined with the limitations of WSN nodes gives unique scenarios for which a commercial all general purpose WSN standard such as ZigBee [8] show its limitations in an industrial setting. [9]

There have been many standards applied for WSN such as Zigbee, ISA100.11a, and WiHART [10]. WiHART has been successful in providing a multi-layered protocol suite which satisfy most requirements in the field. For this reason, this project will use WiHART as a point of reference in regards to industrial standards.

2.6.2 The History of Industrial WSN MAC protocols

In a general all purpose WSN one of the major aspects is to conserve the power of the nodes to increase the node life-time. Thus, the two contention based protocols S-MAC and T-MAC were proposed.

S-MAC [11] introduced an active-sleep cycle scheme to force nodes to sleep in order to conserve energy. It also used various collision and overhearing techniques in order to improve the sleep cycle. The results were low power consumption, but the latency was increased and the throughput was low. S-MAC also broadcasts data packets, which increases collisions and overhearing which wastes power. T-MAC[12] later improves upon S-MAC by introducing an adaptive sleep cycle.

Contention for medium is one of the main sources of delay, thus TDMA-based MAC protocols have been proposed, as well as combined hybrid schemes using TDMA and CSMA in conjunction. Time-slots are given in order to prevent nodes to interfere with each other and guarantee timely delivery of packets.

TDMA has the advantage of collision free medium access, however disadvantages include clock drift and decreased throughput due to idle slots. It also has major problems with synchronization of nodes and adapting to topology changes. Another problem that needs to be solved is the funnel effect, where the node close to the sink has to process larger volumes of packets.

Tree-MAC[13] introduces a tree-strucutre and the SuperFrame with frames which are assigned from parent to children. This concept eliminates verticle two-hop interference and enables slot re-use.

I-MAC was introduced to deal with the slot-reuse issue which causes interference, and suggests bi-directional links for reliability and uses a spare time utilization scheme instead of slot reuse.

2.6.3 Defining Key Industrial Parameters

This section will mention what this paper considers important parameters for a WSN protocol to measure when used in an industrial context-sensitive safety monitoring application. Many of the definitions have been taken from [9] which defines Reliability, Latency, Sensor Data Update Rates, Wireless Transmission Range, Power Consumption, and Integration with PCDA Systems as key parameters.

The paper [9] has developed the following requirements as industrial requirements. The definitions are also lifted from the paper and we will be using these parameters for our analysis of I-MAC and BigMAC in how they satisfy these requirements.

– Reliability

– Latency

– Sensor Data Update Rates

– Wireless Transmission Range

– Power Consumption

The paper will also consider two other important requirements, scalability, adaptability, interference, implementation complexity and flexibility. Although important, security is left out of the scope of this thesis.

– Scalability

– Adaptability

– Interference

– Implementation Complexity

– Flexibility

Reliability

When a node-A sends a packet to node-B, node-B will reply with an ACK to acknowledge that the packet has been received. If a packet is lost, then after some timeout period of not receiving the ACK node-A will re-transmit the package. This ensures that the packet will eventually arrive at node-B.

Reliability in WSN can be quite tricky. Estimating link quality can be very difficult due to the asymmetrical nature of transmitting nodes, and thus invalidates many assumptions made about reliability in other environments.

It is important to note that Reliability can be end-to-end and link-by-link.

Reliability can be measured as the percentage of data that reaches its destination, PDR (Percentage Data Reached).

Latency

Latency is defined as time-delay, namely the time it takes from the data from the original sender to reach the final destination. Link-Quality is an important factor when addressing latency, as a poor link introduces re-transmissions which results in higher latency.

It is easy to understand that if there is a high latency in a WSN network for industrial safety monitoring applications that the consequences can be severe.

Sensor Data Update Rates

The frequency of sensor data updates increases the power consumption, thus a trade-off is needed.

Wireless Transmission Range

One of the key problems of WSN is the density of nodes, due to sensor ranges. This creates a problem with the transmission range, as too high of a range will create more signal collisions and overhearing.

Power Consumption

Due to the cumbersome nature of replacing batteries, extending the lifetime of a node by reducing power consumption is of high priority.

Scalability

The scalability of a node is how well the protocol scales with increased nodes before degradation occurs. Also, how it functions in different environments are of interest.

Adaptability

The nature of industrial environments are that of high interference and signal blocked leading to a very dynamic environment. How adaptable is the protocol to changes is in network size, node density, topology changes. How fast is the node join time, and flexibility of nodes losing parents.

Interference

Protocols require recovery mechanisms or active ways to combat interference. In industrial applications interference can be very high.

Flexibility

What are the limitations of the setup for the protocol. What are the flexibility in terms of what kind of application it can be used for.

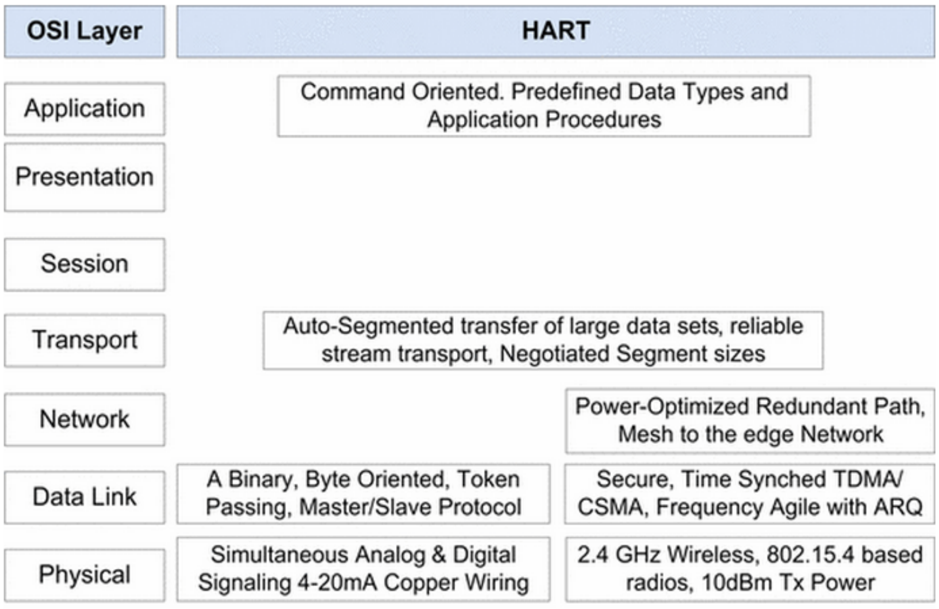
Implementation Complexity

How complicated is it to set up the WSN network using the protocol.

Chapter 3 WirelessHART

3.1 Introduction

WirelessHART is a protocol suite developed for WSN based on the wired HART protocol suite. The HART foundation is responsible for the standardization of this protocol and it is the first standardized open industrial WSN protocol dedicated to industrial process control. See figure 3.1 and 3.2 for the structure of the layers of WiHART.



OSI Layer	HART	
Application	Command Oriented. Predefined Data Types and Application Procedures	
Presentation		
Session		
Transport	Auto-Segmented transfer of large data sets, reliable stream transport, Negotiated Segment sizes	
Network		Power-Optimized Redundant Path, Mesh to the edge Network
Data Link	A Binary, Byte Oriented, Token Passing, Master/Slave Protocol	Secure, Time Synched TDMA/ CSMA, Frequency Agile with ARQ
Physical	Simultaneous Analog & Digital Signaling 4-20mA Copper Wiring	2.4 GHz Wireless, 802.15.4 based radios, 10dBm Tx Power

Figure 3.1: WiHART Layers. [10]

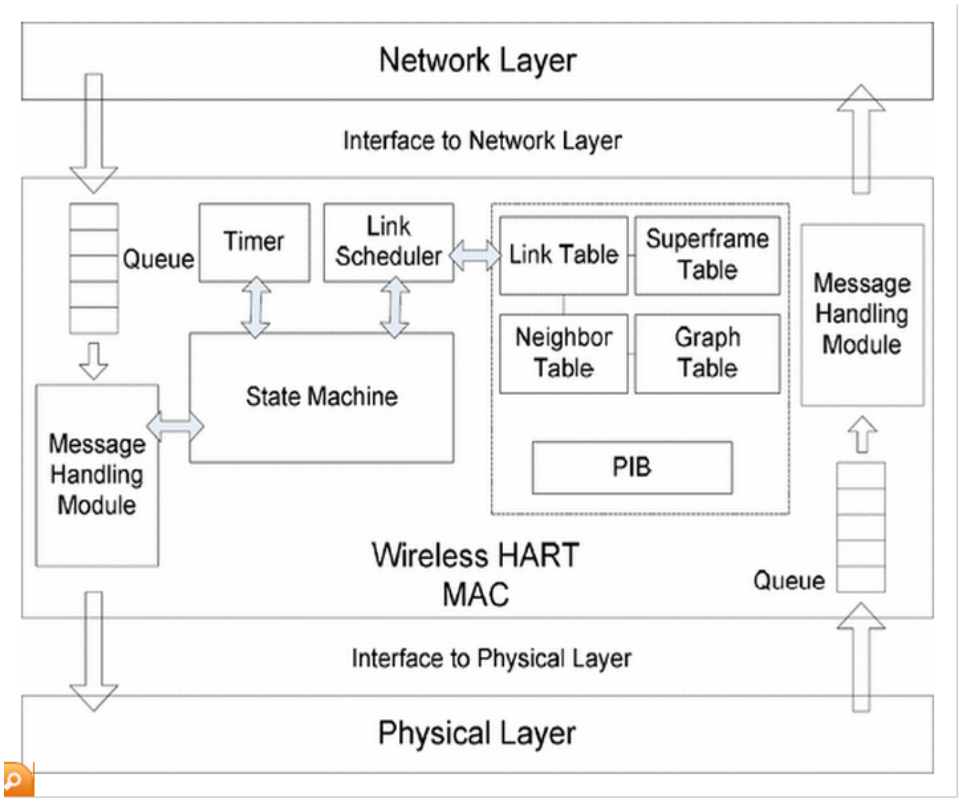


Figure 3.2: Network, MAC and Physical . [10]

Most of the changes from wired HART have been done on the physical layer, the MAC layer, and the network layer. The focus of WiHART is to make the protocol viable in industrial process control environments which are known to have heavy interference and have strict timing requirements. WiHART uses various mechanisms which will be discussed in this chapter to provide reliable throughput with a guaranteed delay and it uses channel hopping to combat interference.

WiHart also provides the network with numerous enhancements in security and reliability. The network inter operates with HART legacy systems and the network is self-organizing and self-healing. Both star and mesh topology are supported, and WiHart includes built-in time synchronization. Time Division Multiple Access (TDMA) is used for slot scheduling and Graph routes are used to route packets.

WiHART introduces the network manager, the security manager and operates through a centralized model. The network manager takes alot of the computational burden away from the network nodes and performs many of the important features. The security manager takes care of the security such as distributing keys and authenticating nodes. It is worth noting that the WiHART standard is a set of schematic plans without detailed implementation, this gives room for different realizations of the protocol suite. For this reason, and due to the fact that WiHART requires strict timing it makes it very difficult to evaluate and test, even in a controlled environment.

WiHART limits the network size between 80 - 100 devices per gateway. This limit makes using a centralized network more efficient and practical. By using TDMA it grants predictability to the communication over the CSMA contention scheme. The next section will discuss the network manager following this the mechanisms and schematics in the IP and MAC layer will be explained.

3.2 Network Manager

The network manager is introduced in WiHART to control the routing and time scheduling of the system in order to avoid collision and ensure data reaches its destination in a timely manner. This allows for simpler network devices that utilize less power due to removing the computational burden from the nodes. The network manager has full view of the topology which it gets from periodic updates from each node. It also receives updates of the nodes neighbors and signal strength. By giving the network manager a global view, it improves the transmission efficiency and the security of the system. The network manager has the responsibility of routing and scheduling.

It utilizes centralized management techniques for communication scheduling and to manage routes. However, WiHART does not define specific algorithms to be used by the network manager to allocate resources or building routing tables.

This section will outline the mechanisms used in the IP and and MAC layer used in WiHART respectively. In the WiHart protocol the route and link scheduling policy is dependent on the centralized network management algorithms, and these algorithms may not be optimal.

The network manager is responsible for all routing and scheduling messages through network information in combination with the reported requirements provided by the devices and applications.The schedule is divided into time slots and transferred from the network manager for individual devices through the gateway and access point.

Three basic "events" in WiHART. That is the advertisements, association and resource allocation. Advertisement, nodes that are in the network send out packets announcing the presence of the network and time synchronization information and the Net ID.

3.2.1 IP Layer

This section will give and overview over the contents of WiHART present within the IP layer. The major focus will be on how WiHART handles the routing and explaining in-depth the inner workings of the network manager as it relates to the IP layer, See figure 3.3.

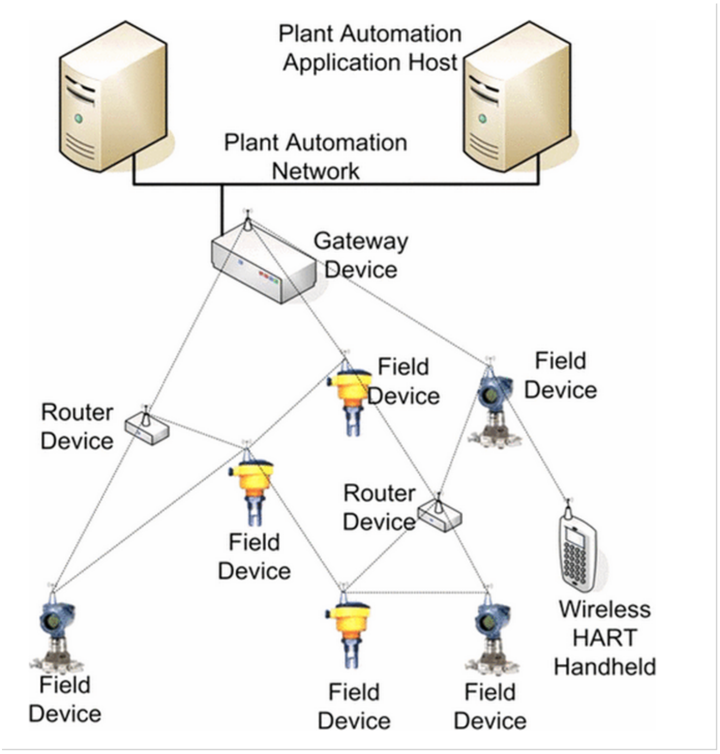


Figure 3.3: WiHART Network Layer. [10]

Source Routing

WiHart makes use of source routing mainly for diagnostic purposes in the network. For routing packets Graph Routing.

Graph Routing

A graph route is a collection of paths that connect devices. The network manager has the responsibility to create the paths and distribute them to the different nodes in the network. The communication schedule will be generated only after the routes are constructed and downloaded to every device connected to the network.

Each node contains a Graph routing table within its MAC layer. Whenever a packet is being sent out, the node checks its graph table and compares it with the graph ID within the header of the packet that is being sent. The correct outgoing interface is chosen based on the graph ID.

Within the WiHART protocol there are three standards for different communication. The broadcast graph which is used to propagate common control messages. The up link graph, used to relay sensor data to the sink. And each node also has a unique down link graph.

.

Graph Routing Algorithm

Although no routing algorithm is specified in the WiHART specification, it is an essential component. A good or bad algorithm may make or break the implementation of the protocol. The role of the routing algorithm is to calculate the Graph Routes to the source that will be distributed to the nodes by the network manager. A poor set of graph routes may result in poor network performance, and a slow algorithm may result in poor network stability and conditions should the need for a full network repair arise.

3.2.2 MAC Layer

Since most of the functionality are moved into the network manager, the main responsibilities of the MAC layer is to store information in its Link table, Superframe Table, Neighbor table and graph table.

The nodes in WiHART contain a link table, a super frame table, a neighbor table and a graph table.

State Machine

The state machine has three primary uses. To signal the XMIT and RECV engines.

Communication Tables

The nodes in WiHART contain a link table, a super frame table, a neighbor table and a graph table. These tables contain information that control the communication of the network nodes. The tables can also be used for reporting statistics to the network manager.

Link Table

Super Frame Table

The main task of the super frame table is to provide information to help with the configuration between neighboring nodes. The table contains four fields, see figure x. The network manager will distribute the super frames depending on network topology.

Neighbor Table

Contains a list of all the neighbors a node has. Will be used as data for the network manager when deciding which route to take.

Graph Table

The graph table is used to guide packets to their final destination. A packet sent through the network will have attached to its header a graph ID. The node will look up the graph ID in its graph table and forward it to the next destination based on this.

Channel Hopping

WiHart can use up to 16 different channels when sending messages, spreading the WiHart communication in all physical active channels in 802.15.4. Every message being sent will pseudo randomly chose one of the 16 channels available to transmit the message. By using this method, the network manager can "blacklist" channels that are prone to interference. The cost of spectrum diversity is increased scheduling complexity.

WiHART is set up so that only one transmission at a time can use a channel in the time slot given across the network. This improves reliability, however reduces throughput, but in a small network the damage is mitigated. This feature also avoids spatial reuse of channels.

TDMA

A link consists of a SuperframeID, source and destination IDs, a slot number referenced to the beginning of the superframe and a channel offset. A link is a transaction that occurs within a cell. WiHART uses TDMA for scheduling. It has a strict 10ms timeslot, time enough for one transmission and acknowledgement between pairs. How and what algorithm is used for scheduling is not stated in the WiHART protocol.

A Super frame is a collection of cells which repeats at a certain interval. Each slot in the superframe is 10ms, see figure 3.4. By adjusting the time slots in such a way that transmitting and receiving into one cell, the adjacent links will never transmit at the same time and the hidden terminal problem is avoided all together.

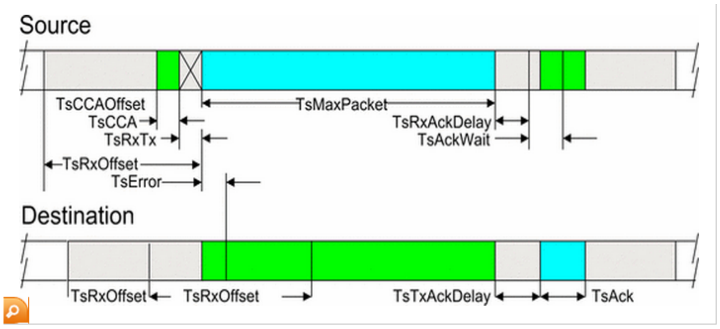


Figure 3.4: Slot Timing. [10]

Time Synchronization

WiHART uses Time Synchronization Mesh Protocol TSMP, and it uses the gateway as the root for the source of time. Some nodes are specified as "Time synch sources", which are used to help with synchronization. See [10] for more information on time synchronization.

Chapter 4
I-MAC

4.1 Introduction

This chapter will introduce in detail the inner workings of the key concepts of I-MAC and its mechanisms and how they work together to support the industrial requirements. I-MAC was designed at Ubicom Lab, School of Computer Engineering and Information Technology, University of Ulsan as proposed in the paper [sauce]. It is a protocol created to be reliable, power balancing, and working with dynamically changing topology in a timely manner for use in small networks consisting of 30-50 nodes[2]. It uses a TDMA scheme over a contention based one, and uses various mechanisms to make TDMA function in a dynamic topology.

4.2 I-MAC Protocol Description

I-MAC uses a tree-topology with TDMA to give every node a unique time-slot where it can transfer and receive data. Before constructing the tree, all the nodes must be synchronized to some global time. After synchronization and tree construction, the protocol schedules time-slots for every node.

There are three stages of the protocol. The first on is the Initial Construction Period (ICP), followed by the Reliable Control Transmission Period(RCTP), Reliable Data Transmission Period (RDTP) and finally the maintenance period (MP). See figure 4.1

The maintenance period performs functions such as time synchronization, tree maintenance and slot scheduling. RCTP and RDTP are periods where control packets and data packets are transmitted respectively.

As seen in figure 4.1, after the first ICP, the RCTP, RDTP and MP periods repeat, this is the protocol cycle.

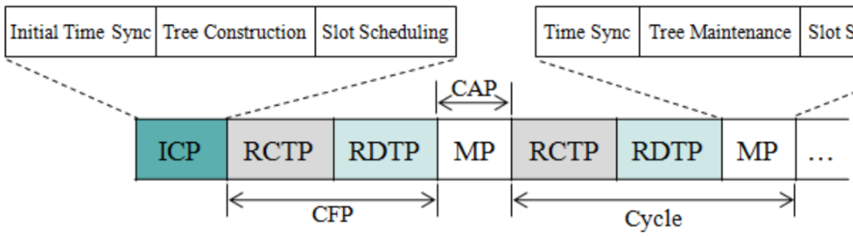


Figure 4.1: I-MAC Stages[2].

4.2.1 Time Synchronization

In a TDMA based protocol it is important for all nodes to synchronize to a global time. In I-MAC time synchronization is performed during the ICP and MP period. The protocol uses Flooding Time Synchronization Protocol [14] which provides comprehensive error compensation and estimates for clock skew.

When the sink wants to initiate time synchronization it broadcasts a synchronization message SYNC.

– SYNC (sinkId, sender, seqNum, globalTime)

  sinkId is the id of the sender, and the sender is the sink or rebroadcasting node. The seqNum is used to deal with redundant messages and globalTime is the estimated time of the sink by the broadcaster.

Each node uses a table for use with linear regression with the pair offset, LT, where LT is the local time. Due to clock drift, the clock offset calculated by comparing local time of the node with the global time from the SYNC message is not constant. Thus, linear regression is used. [2] presents two equations to calculate the offset and estimate global time of the sink.

$$offset = of\bar{f}set + skew * (LT - L\bar{T}) \tag{4.1}$$

$$GT = LT + offset. \tag{4.2}$$

4.2.2 Tree Construction and Maintenance

After the nodes have synchronized time, the tree construction can being. However, in order to optimize the reliability of the tree-structure, it must determine the link quality and identify if the link is bi-direct reliable links.

After the tree has been constructed, the next step is to calculate the slot demand of each node.

Link Quality

In order to provide reliability to the protocol, a simple reliability mechanism using Request to send(RTS) and Clear to send(CTS) together with an acknowledgement message is used. However, due to the difference in transmission range between nodes and interference, it is important to identify the link quality. If the link quality is bad, it will result in poor reliability and higher latency.

I-MAC needs to create a bi-directional tree in order to use the RTS/CTS mechanism. When constructing the tree it is important to make sure the links connecting the tree is of high quality. This is done by using metrics to identify the quality of the link. In [] they find that combining three metrics received signal strength indicator (RSSI), link quality indicator (LQI) and packet reception rate(PRR), they can gain quick and reliable assessment of the quality of the link (linkq).

$$linkq = R\bar{S}SI + L\bar{Q}I \tag{4.3}$$

Identification of Bi-Direct reliable links

In WSN a bi-directional link is commonly assumed. However, in a real life scenario, different antenna/transmission power, intereference, blocking and fading etc can disrupt the bi-directionality of the topology. Therefore, it is important to identify that the links are indeed reliable and bi-directional.

IMAC tries to avoid high overhead, thus neglects to use control messages to confirm the bi-directionality. In IMAC each node has a LST lnk state table, see figure 4.2, containting the status if a neighbour node is bidirectiaonlly stable. When the neighbor sends messages, it can overhear and update its links state list. If no message is received or overheard over a period of time, the link state table is updated to uncertain.

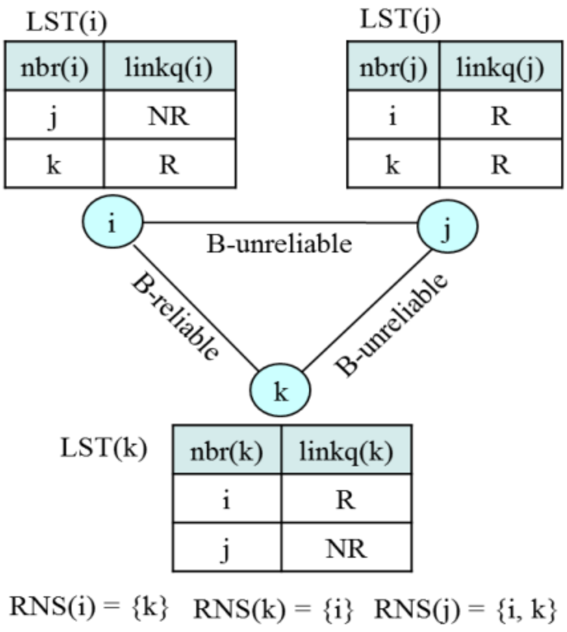


Figure 4.2: Example of a link-state table [2].

Tree Construction with Time Synchronization

The three synchronization process uses the three messages:

- Tree Construction Request TCR(sinkId, seqNumber, RNS, globalTime)
- Join Request J-REQ(sender, receiver, depth, RNS, globaTime)
- Join Response J-RES(sender, receiver, depth, RNS, globalTime)

The TCR is broadcasted to initiate the tree construction process and the orphans use J-REQ to join and J-RES to respond to the request. If another node overhears the J-REQ, then that node itself can send a J-REQ to join the sending node. A join delay timer is used to prevent collision.

To avoid creating an inefficient tree, the B-reliability and hop-distance to the sink is used as a metric to decide which node to join, if it overhears more than one J-REQ.

Tree Maintenance

The I-MAC paper defines a Data Acquisition Ratio (DAR).

$$DAR = nNodesSent \div nNodes \tag{4.4}$$

This variable is used and compared to a threshold variable DAR_Threshold which decides that the topology is damaged enough to warrant reconstruction.

When the a link breaks occur between parent and child, and the child nodes need to change parent, there will be changes to the slot demands of their ancestors. This makes it too complicated to perform a local tree repair and reconstructing the three is too costly. Thus, the three is not reconstructed until reaching the DAR_Threshold. I-MAC introduces the spare time utilization scheme to account for salvaging lost packets and keeping reliability.

In an environment with high frequency of link breaks, the performance of the protocol might decline quickly. One weakness of I-MAC is when a parent node dies, the children cannot rejoin the network until the next cycle, as all the time slots have already been assigned. In addition to this, the damage to the three must have reached a certain threshold for it to warrant reconstruction and rescheduling. Also, change of ancestors requires change to new slot demand. In networks with high link breakage this can accumulate very quickly and result in severe performance degradation.

4.2.3 Slot Scheduling

One of the advantages that I-MAC gains by using TDMA is eliminating contention between nodes. Contention has shown to be one of the main contributors to delay[2], however there are trade offs by using TDMA. TDMA suffers low throughput compared to a contention based scheme in low traffic networks, due to idle time slots. It also requires global synchronization and needs to be bidirectional in order to be reliable, which increases the complexity of the protocol.

Contention based schemes also suffer from the funneling effect[2], so the nodes closer to the sink has higher power consumption due to the higher traffic intensity, and thus collisions and loss of packets. I-MAC uses the DSA algorithm in the scheduling [2], this reduces the load balance on the nodes closer to the sink.

I-MAC uses two algorithms to calculate and assign slot demands. These are the Slot Demand Calculation (SDC) algorithm and the Slot Demand Assignment (SDA) algorithm. I-MAC requires both control and data slots to be assigned.

Slot Demand Calculation Algorithm

In this phase the slot demand for data and control slots are calculated from the leaf nodes and up the tree. A Slot Demand Calculation message (SDC_MSG) is used to report to the parent the slot demand of a node.

$$SDC\_MSG = (Ci, Di, T(i)), \tag{4.5}$$

where Ci and Di are the number of control slots and data slot that node i demands, and T(i) is the set of nodes that belong to a tree whose root is under node i.

Control packets are sent from the sink down the tree, and for this reason the leaf nodes requires no control packets, as they have no children to forward them to. However, they do require to send one data packet to their parents. So the leaf node demand is zero control slots and one data slot, and will send SDC_MSG(0, 1, 1) to its parent.

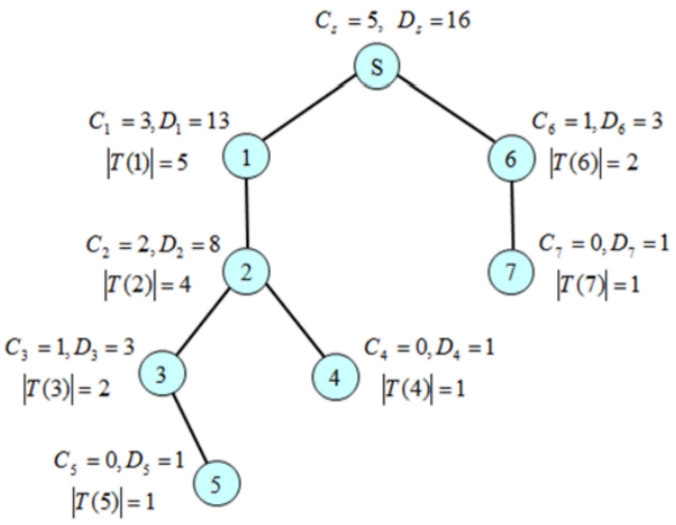


Figure 4.3: Slot demand calculation in tree-topology. [2]

For the intermediate nodes i, the data slot demand is calculated by the adding together the children's data slot demands and adding the number of nodes from the sub-tree T(i). So node 2 in figure 4.3 has 4 nodes in its sub-tree T(2) = 4, and its children require data slots 3 and 1 respectively, thus the data slot requirement of node two is 8.

The control slot demand of a node is the sum of the control slot demands of its children plus the number of control slots it needs to forward one control message from the sink to its children, in one cycle.

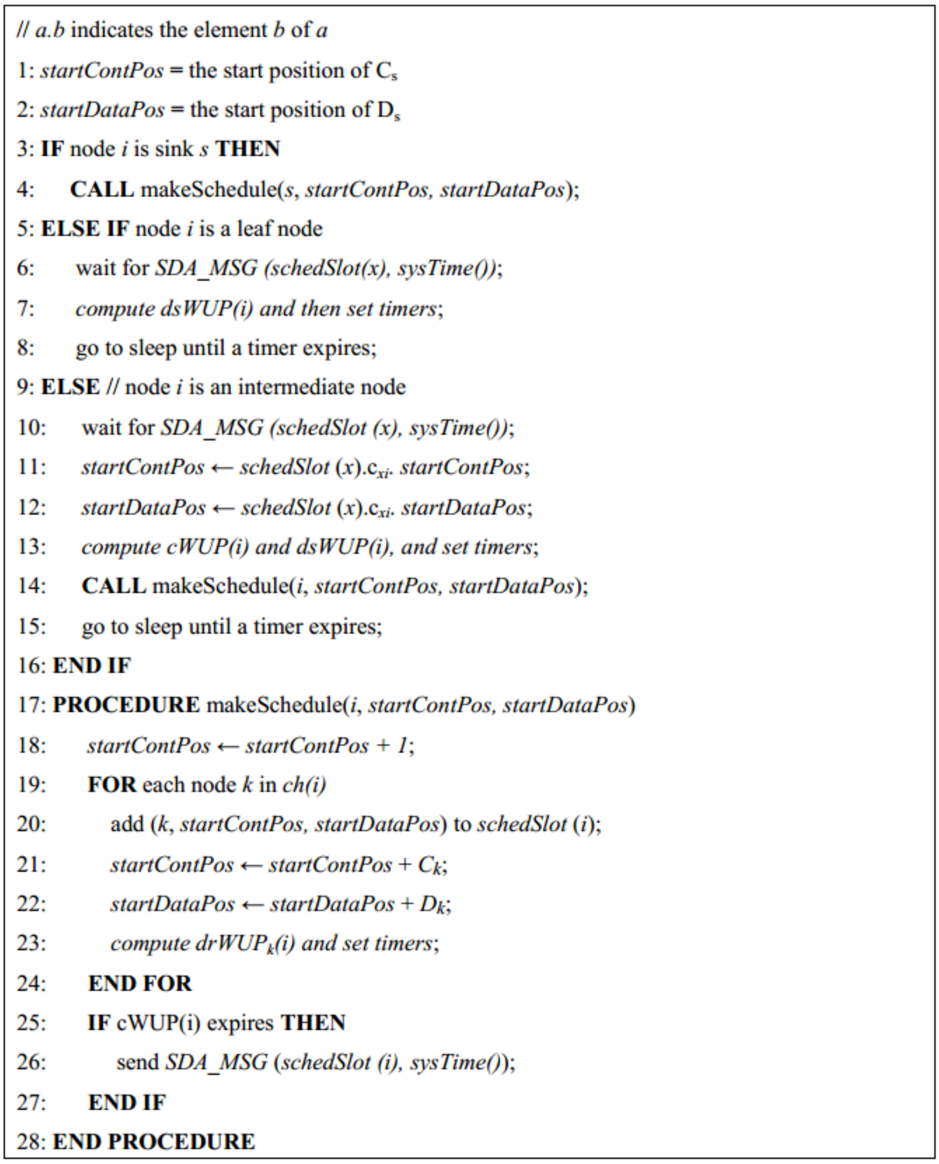
Slot Demand Assignment and Power Control

The SDA algorithm is performed from the sink down to the leaf nodes. It uses a Slot Demand Assignment message (SDA_MSG).

$$SDA\_MSG = (scheduSlot(i), sysTime(), T(i)), \tag{4.6}$$

where Ci and Di are the number of control slots and data slot that node i demands, and T(i) is the set of nodes that belong to a tree whose root is under node i.

It is performed after the completion of the SDC algorithm, and uses a slotted broadcasting mechanisms to do distribute the slot assignments. See 4.4 for details on the algorithm.



```
// a.b indicates the element b of a
1: startContPos = the start position of $C_s$
2: startDataPos = the start position of $D_s$
3: IF node i is sink s THEN
4:     CALL makeSchedule(s, startContPos, startDataPos);
5: ELSE IF node i is a leaf node
6:     wait for SDA_MSG (schedSlot(x), sysTime());
7:     compute dsWUP(i) and then set timers;
8:     go to sleep until a timer expires;
9: ELSE // node i is an intermediate node
10:    wait for SDA_MSG (schedSlot (x), sysTime());
11:    startContPos ← schedSlot (x).$c_{xi}$. startContPos;
12:    startDataPos ← schedSlot (x).$c_{xi}$. startDataPos;
13:    compute cWUP(i) and dsWUP(i), and set timers;
14:    CALL makeSchedule(i, startContPos, startDataPos);
15:    go to sleep until a timer expires;
16: END IF
17: PROCEDURE makeSchedule(i, startContPos, startDataPos)
18:    startContPos ← startContPos + 1;
19:    FOR each node k in ch(i)
20:        add (k, startContPos, startDataPos) to schedSlot (i);
21:        startContPos ← startContPos + $C_k$;
22:        startDataPos ← startDataPos + $D_k$;
23:        compute $drWUP_k(i)$ and set timers;
24:    END FOR
25:    IF cWUP(i) expires THEN
26:        send SDA_MSG (schedSlot (i), sysTime());
27:    END IF
28: END PROCEDURE
```

Figure 4.4: The SDA Algorithm [2]

Chapter 5 BIG-MAC

5.1 Introduction

This chapter will introduce Skewed Big-Slot Scheduling based protocol for Real-time and Reliable Data Transmission in Wireless Networks. The protocol will be refereed to BigMAC as it focuses on big-slot scheduling.

The goal of BigMAC is similar to that of I-MAC, they both focus on improving the reliability in WSN applied in the field of time-constraint monitoring and control application. One of they key differences it that I-MAC uses a pure TDMA approach, while the concept in BigMAC is to use a hybrid scheme TDMA/CSMA based approach which will be explained in detail later. BigMAC aims to improve upon some of the drawbacks that TDMA introduces by using big-slot scheduling to reduce complexity and increase adaptability and scalability.

The concept of BigMAC is to use TDMA to provide time constraint to the data delivery, and at the same time use a contention based scheme CSMA locally within a "Big-Slot" to allow for flexibility within the same node depth. The Big-Slot is allocated in a distributive manner and all nodes on the same level share the same Big-Slot.

This chapter will introduce the key concepts, protocol structure and mechanisms of BigMAC and how they relate to the industrial requirements. The following chapter will then compare and analyze the difference between I-MAC and BigMAC.

5.2 BigMac Protocol Description

The paper [3] defines a big-slot and a superframe(SF) as the following:

Definition: Big-Slot
A big slot is a time span that all nodes at the same depth share to receive data packets

from their children, and transmit their data packets to their respective parents using CSMA. A big slot allocated to the nodes at depth i is denoted as BS(i), the portion of the BS(i) for the nodes at depth i to receive data packets form their children is denoted by

$$BS^{RX}(i) \tag{5.1}$$

and that of the BS(i) for the nodes at depth i to transmit data packets to their respective parent is denoted as

$$BS^{TX}(i) \tag{5.2}$$

.

Definition: Superframe
A Superframe(SF) is given the sum of the transmission portions of all the big-slots allocated to the nodes at different depths as follows.

$$SF = \sum_{i=2}^{H} BS^{TX}(i) \tag{5.3}$$

Similar to I-MAC the BigMAC protocol has a Initial Construction Period ICP, followed by a Reliable Data Transmission Period and a Maintenance period. The RDTP and MP repeat like in figure 5.1. Notice that BigMAC does not have its own period for control packets like RTCP in I-MAC.

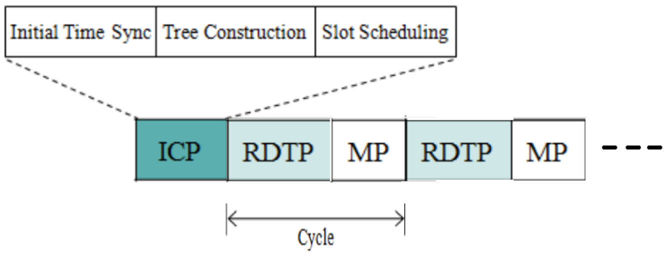


Figure 5.1: The phases in BigMAC protocol

Time synchronization, tree construction and slot scheduling all occur during the ICP period. If necessary, during the MP these actions will be performed as required.

During the slot scheduling, the Big-Slot is distributed to all the nodes at each depth. The Big-Slot works to constrain the time of transmission for the nodes within a time slot. However, at the same depth the nodes use CSMA to compete for the time slots. This is unlike I-MAC which is a pure TDMA approach.

5.2.1 Tree Construction and Maintenance

Link Quality Estimation and Bi-Directional Reliable Link

Similar to I-MAC, BigMac uses a bi-directional tree for communication. The same problem of identifying if a link is reliable occurs here. BigMAC uses the same metric as I-MAC with the equation given below.

$$linkq = R\bar{S}SI + L\bar{Q}I \tag{5.4}$$

BigMAC uses a modified method for detecting Bi-directional reliable links. For more details see the paper [3].

Reliable Tree Construction

BigMAC tries to construct trees that consist of B-reliable links only. This will enhance the reliability of the the network.

The sink will broadcast a sync message, and when a node receives it, they rebroadcast the Sync and perform initial time synchronization. The modified Flooding Time Synchronization Protocol is used.

After synchronization is complete, the sink will initiate tree construction by sending a Tree Construction Request.

Most of the steps for creating a tree is very similar to the steps found in I-MAC. There are however some modifications, but these are not important to the scope of the project.

5.2.2 Slot Scheduling

BigMAC has a significantly less complex slot scheduling scheme than I-MAC. This is due to limiting the TDMA to a global big-slot on each node depth. This allows for flexibility in the time-slots, since they are contended for. Some of the benefits are, no idle slots and wasted throughput, parallel transmission of nodes that do not interfere with each other, and the ability for a node to change parent on the fly.

Wait Time Generation Function

Since BigMAC uses a local contention based scheme among nodes on the same level, then there must be some way of determining of how long the contention period should be. The wait time generation function will be used for helping making an efficient packet scheduling scheme.

$$WTime(d) = W1 * a^{d-1} \tag{5.5}$$

Where WTime(d) is the time that the node at depth d must wait before transmitting data to its parent, W1 is the Superframe SF, and a is the range of the base.

Big-Slot Length

Due to the nature of the tree topology, it can be surmised that the deeper the depth in the tree, the larger the BS length must be, due to it being shared for all nodes on each depth.

Big-Slot Assignment Algorithm

After time synchronization each node performs slot scheduling in a distributive manner using equation [5.5].

Using sTime as the start time of a cycle, (assuming it is synchronized) then a node at depth i will have the big slot schedule given by the follow equations:

$$RxTime(i) = sTime + WTime(i + 1) \tag{5.6}$$

$$TxTime(i) = sTime + Wtime(i) \tag{5.7}$$

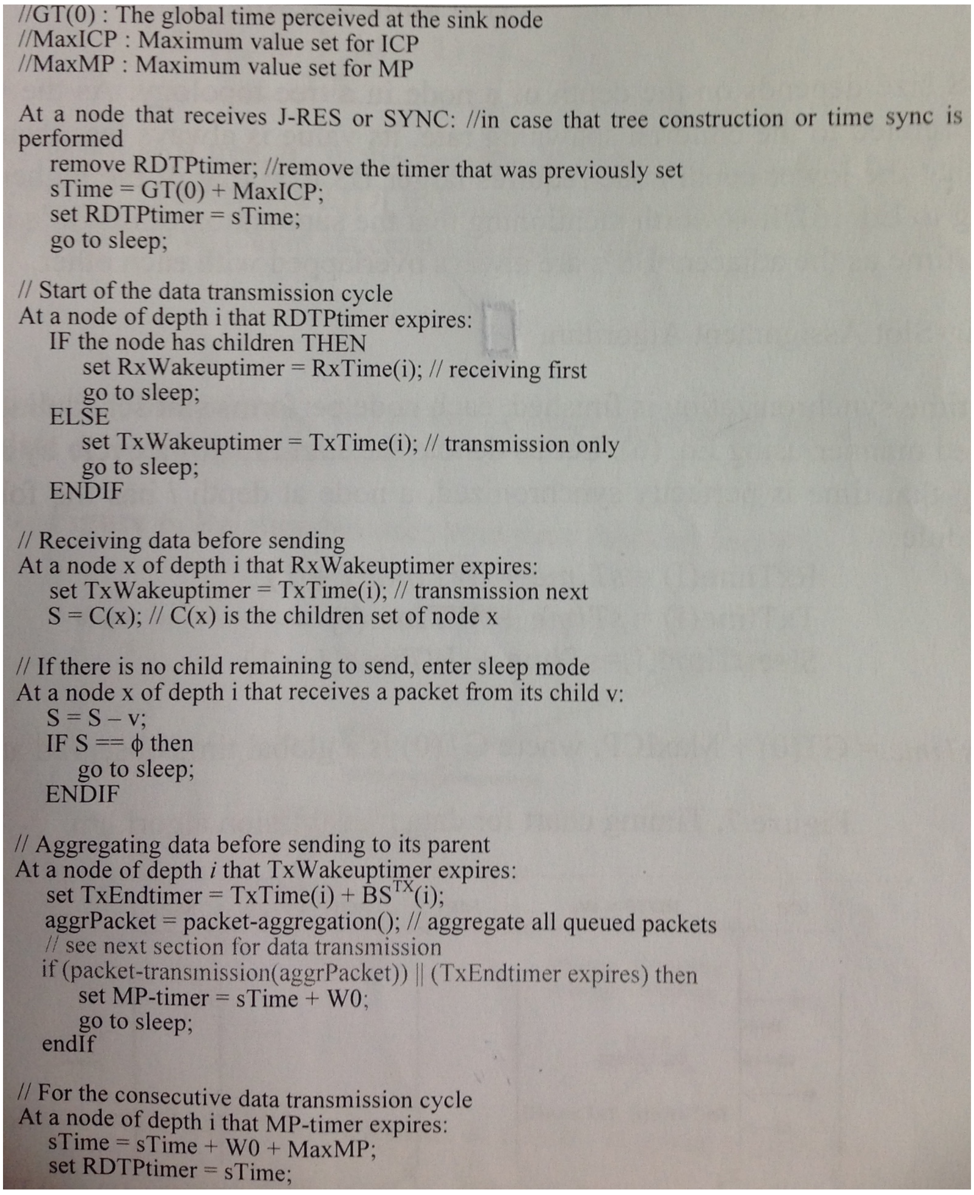
$$SleepTime(i) = sTime + Wtime(i - 1) \tag{5.8}$$

Where,

$$sTime = GT(0) + MAXICP \tag{5.9}$$

Where MAXICP is the maximum time span that every node finishes joining the tree.

These equations are used in the BigMAC packet scheduling algorithm presented in figure 5.2.



```
//GT(0) : The global time perceived at the sink node
//MaxICP : Maximum value set for ICP
//MaxMP : Maximum value set for MP

At a node that receives J-RES or SYNC: //in case that tree construction or time sync is
performed
    remove RDTPtimer; //remove the timer that was previously set
    sTime = GT(0) + MaxICP;
    set RDTPtimer = sTime;
    go to sleep;

// Start of the data transmission cycle
At a node of depth i that RDTPtimer expires:
    IF the node has children THEN
        set RxWakeuptimer = RxTime(i); // receiving first
        go to sleep;
    ELSE
        set TxWakeuptimer = TxTime(i); // transmission only
        go to sleep;
    ENDIF

// Receiving data before sending
At a node x of depth i that RxWakeuptimer expires:
    set TxWakeuptimer = TxTime(i); // transmission next
    S = C(x); // C(x) is the children set of node x

// If there is no child remaining to send, enter sleep mode
At a node x of depth i that receives a packet from its child v:
    S = S – v;
    IF S == $\phi$ then
        go to sleep;
    ENDIF

// Aggregating data before sending to its parent
At a node of depth i that TxWakeuptimer expires:
    set TxEndtimer = TxTime(i) + $BS^{TX}(i)$;
    aggrPacket = packet-aggregation(); // aggregate all queued packets
    // see next section for data transmission
    if (packet-transmission(aggrPacket)) || (TxEndtimer expires) then
        set MP-timer = sTime + W0;
        go to sleep;
    endIf

// For the consecutive data transmission cycle
At a node of depth i that MP-timer expires:
    sTime = sTime + W0 + MaxMP;
    set RDTPtimer = sTime;
```

Figure 5.2: BigMAC Packet Scheduling Algorithm. [3]

RxTime is used to wake up and listen for receiving data packets, then immediately sleep again until TxTime, where the node wakes up, aggregates all received packets and send the packet further up the tree to its parent using CSMA.

Since BigMAC aggregates alot of the packets, it has the advantage of removing many control messages and headers of data packets. This in turn reduces the total power consumption. Nodes close to the sink benefit immensely from this as well due to the power balancing issue.

Data Transmission and Tree Maintenance

During the MP, orphan nodes will initiate the tree-joining process by broadcasting JREQ. It will receive the JRES from nodes with depths less than itself and and be given a parent. However, different from I-MAC, if it receives more than one JRES, it can take other nodes as auxiliary parents, which can be used as replacement should its parent die.

A node must transmit its aggregated data between the TxWakeuptimer and TxEndtimer using CSMA. If, by this time it was not able to send it will give up sending. If the first try ends in failure, it will wait a random back-off time before trying to resend. If, for some reason the second try fails, it will determine that the link has been broken. The next step is to change its parent to one of the auxiliary ones and try again. If it is successful the parent will take the node as its new child.

Chapter 6

Theoretical Comparison of I-MAC and BigMAC

This chapter will compare the two protocols I-MAC and BigMAC by comparing the mechanisms and key principles used. The two protocols may have different pros and cons, and this chapter will try to identify if there are certain scenarios where one protocol might outperform another.

6.1 I-MAC and BigMAC

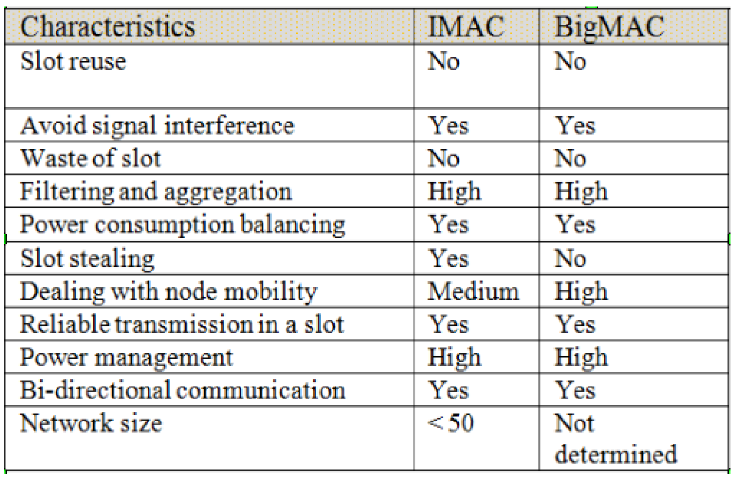
6.1.1 Key Concepts

I-MAC uses a TDMA based approach versus BigMACs hybrid scheme of TDMA/C-SMA. Many of the main differences in I-MAC and BigMAC has been summarized in table 6.1.

The main reason for I-MAC to use TDMA was to guarantee timely delivery, and at the same time eliminate much of the latency which comes from having a contention based scheme. The downside is the complicated slot scheduling and loss of throughput due to idle slots in low traffic networks.

BigMac uses a hybrid approach which maintains the timely deliver of TDMA with a much simpler slot scheduling mechanism. It also has the capability of parallel transmissions of nodes in the same level that do not interfere with each other.

Another difference between I-Mac and BigMAC can be seen in the protocol flow. They both have the phases ICP, RTDP and MP, however I-MAC also includes a phase RTCP where all data transmission stops and only control packets are in play. This begs the question, if there are no data packets in play in this period, then during this period the protocol is not reliable. This phase occurs every cycle, and during this period of shuffling control messages it consumes power and causes latency in data delivery. BigMAC has no such period, and it would be interesting to investigate the performance hit of this period.



Characteristics	IMAC	BigMAC
Slot reuse	No	No
Avoid signal interference	Yes	Yes
Waste of slot	No	No
Filtering and aggregation	High	High
Power consumption balancing	Yes	Yes
Slot stealing	Yes	No
Dealing with node mobility	Medium	High
Reliable transmission in a slot	Yes	Yes
Power management	High	High
Bi-directional communication	Yes	Yes
Network size	< 50	Not determined

Figure 6.1: Summary of Key Features.

6.1.2 Time Synchronization

6.1.3 Tree Construction and Tree Maintenance

In general both I-MAC and BigMAC both use tree-topology and the tree-construction works very similarly in both protocols. However, the difference in how they use time-slots provides them with very different recovery mechanisms.

Recovery Mechanism

BigMAC was designed to improve the adaptability to the dynamic environments where link breaks occurs often. BigMAC handles it by using a BigSlot with secondary parents as mentioned in the previous chapter, is very resilient to topology changes.

The recovery mechanism in BigMAC should theoretically give it an edge over I-MAC in a topology where link breaks occur often. BigMAC will outperform I-MAC, due to its costly tree rebuild and re-scheduling.

The mechanisms in I-MAC to provide adaptability has weaknesses. First of, due to the complex nature of tree-reconstruction and re-scheduling they introduced the Spare Time Utility mechanism. However, it cannot guarantee to always save the packet if the time-slot is occupied. This may also lead it to potentially waste energy if it in unsuccessful attempts. This might lead to a scenario in which a sub-tree in an area with high interference get stuck indefinitely in a position where the the links break and are reliant on the recover mechanism since the DAR_threshold has not been reached. If there are no free time-slots for the mechanism to steal, then the spare time recovery mechanism might fail to deliver packets, which means there is no timely guarantee for the arrival of packets for that sub-tree. This weakness may reduce I-MACs application in very specific cases.

6.1.4 Slot Scheduling

The complicated slot scheduling presented in I-MAC leads it to degrade quickly as the number of nodes grow. As the number of nodes grows, the number of time slots needed to be allocated grows significantly, and including the complexity of the slot scheduling makes the scaling a nightmare.

BigMAC however, has no limit to the number of nodes, due to its flexible nature. For this reason, the scaling of number of nodes has no theoretical bound. If however, the depth increases in BigMAC, then the big slot wait time might be longer than the contention time, thus introducing more latency than in I-MAC and adding a limiting factor on BigMAC on node depth. This creates a flexibility problem when distributing nodes, as nodes should be distributed in a manner where the lowest amount of depth is present.

Slot-Reuse

I-MAC sacrifices re-usable slots channel efficiency and the ability so send parallel transmissions through nodes that do not interfere with each other. This is a very big strength in a stable network as all nodes have a time to send their data. However, it affects the flexibility of the protocol as it suffers from low throughput in smaller networks. The mechanisms in BigMAC still ensures timely delivery, but also keeps the ability to perform parallel transmissions since the nodes on the same depth use contention. This is also the reason it scales well, since a big slot is shared, only the nodes that have something to send will contend for the big slot. In I-MAC, regardless each node will have its own time slot to send, meaning as the number of nodes grow, so do the required slots. Also, idle slots will be wasted throughput.

6.1.5 Interference

Both I-MAC and BigMAC have recovery mechanisms for when link break occurs. However, both of these protocols lack any mechanism that actively combats interference like WiHART. WiHART uses blacklisting and uses frequency hopping to avoid staying on frequencies that are prone to interference. This is a very useful technique in a dynamic environments.

Chapter 7 Discussion

7.1 I-MAC vs BigMAC

This paper has analyzed the difference between the two protocols on a theoretical level. Some of the findings seem to imply that both protocols have its uses in different applications and scenarios however, BigMAC has yet to be implemented in a real device, and thus further testing is required.

Both protocols handle the industrial requirements quite well. They are both timely and reliable while conserving power. The major area where they differ is within scalability, adaptability and flexibility.

BigMAC has its main advantage in its resilience in dynamic topology, while I-MAC performs well in a stable topology as well as having some mechanisms to provide recovery in high interference environment. There is however, no information or metric on how much interference a general industrial environment has. Some kind of mapping of interference in the different application areas could be useful to determine the validity of both protocols.

In the case of I-MAC and BigMAC there is a certain trade-off in scalability. I-MAC scales poorly with the increases of number of nodes, while BigMAC scales poorly if the node depth becomes too large. However, this trade-off maybe application specific depending if the usage requires a distribution such that the node depth increases, or if it requires a higher node count.

BigMAC allows for parallel transmission, and in theory the increase in node depth should give more parallel transmissions. However, due to the big-sloth size increase as node depth increases, that increase in parallel transmissions might be rendered moot. If the depth is high enough, I-MAC will outperform BigMAC due to the big slot size increase.

7.2 I-MAC reliability

I-MAC has a RCTP period where only control packets are transmitted. This period occurs every cycle, and it can be argued that during this period I-MAC is not reliable.

7.3 Interference

Both protocols lack an active way of fighting against interference such as WiHART. However, perhaps the recovery mechanisms presented these protocols viable enough by themselves. WiHART is after all a multi-layered protocol with a complicated network manager. Maybe introducing a similar scheme to I-MAC and BigMAC will only serve to increase complexity and power consumption.

7.4 Multi-Layered Protocols

Recently, there has been a surge of arguments for using multi-layered protocols such as WiHART over an traditional layered approach. The reasons are that they can share different information among the different layers. This can help with the optimization of the network and also adapting to changes in the environment. However, such protocols are very complex to design and implement.

Chapter 8
Further Work

Most of the work presented is only a theoretical analysis. For future work the theoretical analysis should be verified with experimentation in real physical devices.

8.1 Implementation and Experimentation

8.1.1 Implementing BigMAC on a real-test bed

BigMAC has not been implemented in a real device. For this reason, in order to get a true comparison of the two protocols, BigMAC should be implemented in a similar way as I-MAC. BigMAC should be implemented using tinyOS and nesC as a platform and use TelosB nodes as its hardware.

8.1.2 Experimental Scenarios

Physical Test-Bed with Stable Links

In this experiment, the performance of BigMac on real devices with stable links will be compared to the performance of I-MAC. From the analysis I-MAC should perform quite well in a stable topology.

There also two scenarios to check within this test-bed. Firstly, according to the analysis I-MAC should have low throughput when the traffic is low due to idle slots, this can be checked and compared to BigMAC. Secondly, varying the number of nodes and the node depth could give some interesting results with regards to scalability.

Physical Test-Bed with Dynamic Links

This experiment will be a scenario where link break occurs often to see how the protocols cope with dynamic environments in real devices. The link breaks can either be simulated in software, or the experimenter can physically shut down selective nodes.

This experiment will test the recovery mechanisms, and give some useful performance metric in this kind of scenario.

Chapter 9 Conclusion

This project has analyzed and compared two WSN MAC protocols I-MAC and BigMac for use in an industrial context.

From the analysis it is clear that both protocols has a use, I-MAC being a good candidate in stable topological, while BigMAC outperforms I-MAC in environments where the topology changes occur often. Both protocols have mechanisms in place to handle the industrial requirements.

BigMAC has yet to be implemented in a physical device. Given that experiments in a real test-bed can yield results that vary significantly from simulation results and theoretical analysis, it is important for future work as performance in a real device may deviate significantly from a theoretical analysis.

References

1 The Hart Foundation, http://en.hartcomm.org/ webpage, 20.09.2014

2 Oh, Hoon, and Phan Van Vinh. "Design and Implementation of a MAC Protocol for Timely and Reliable Delivery of Command and Data in Dynamic Wireless Sensor Networks." Sensors 13.10 (2013): 13228-13257.

3 Hoon Oh and Abul Kalad Azad. Skewed big-slot scheduling based protocol for real-time and reliable data transmission in wireless sensor networks DRAFT. 2014

4 Yong-Min, Liu and Shu-Ci, Wu and Xiao-Hong, Nian. The architecture and characteristics of wireless sensor network, IEEE. 2009

5 Simulating a Wireless Sensor Network, http://vlssit.iitkgp.ernet.in/ant/ant/8/theory , webpage, 20.09.2014

6 TelosB Mote, http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf , webpage, 20.09.2014

7 TinyOS, http://www.tinyos.net , Webpage, 20.09.2014

8 Kinney, Patrick. "Zigbee technology: Wireless control that simply works." Communications design conference. Vol. 2. 2003.

9 Radmand, Pedram and Talevski, Alex and Petersen, Stig and Carlsen, Simon. Comparison of industrial WSN standards. 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST), IEEE, Dubai, United Arab Emirates. 2010

10 Song, Jianping and Han, Song and Mok, Aloysius K and Chen, Deji and Lucas, Mike and Nixon, Mark. WirelessHART: Applying wireless technology in real-time industrial process control. 2008

11 Ye, Wei, John Heidemann, and Deborah Estrin. "Medium access control with coordinated adaptive sleeping for wireless sensor networks." Networking, IEEE/ACM Transactions on 12.3 (2004): 493-506.

12 Van Dam, Tijs, and Koen Langendoen. "An adaptive energy-efficient MAC protocol for wireless sensor networks." Proceedings of the 1st international conference on Embedded networked sensor systems. ACM, 2003.

13 Song, Wen-Zhan, et al. "TreeMAC: Localized TDMA MAC protocol for real-time high-data-rate sensor networks." Pervasive and Mobile Computing 5.6 (2009): 750-765.

14 Maróti, Miklós, et al. "The flooding time synchronization protocol." Proceedings of the 2nd international conference on Embedded networked sensor systems. ACM, 2004.