P-1822

# LEEMAC

## A PROJECT REPORT

*Submitted by*

| | |
|---|---|
| **A.C.RAJ KUMAR** | **71203104029** |
| **R.VINOD** | **71203104057** |
| **M.V.DINESH KUMAR** | **71203104501** |

*in partial fulfillment for award of the degree*

*of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE**

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL 2007**

P-1822

---

# ANNA UNIVERSITY: CHENNAI 600025

## BONAFIDE CERTIFICATE

Certified that this project report **"LATENCY AND ENERGY EFFICIENT MAC PROTOCOL"** is the bonafide work of **"A.C.Raj Kumar (71203104029), R.Vinod (71203104057) and M.V.Dinesh Kumar (71203104501)"**, who carried out the project work under my supervision.

(Signature)
Dr.S.Thangasamy

Head of the Department
Dept. of Computer Science & Engg.
Kumaraguru College of Technology.
Chinnavedampatti P.O.
Coimbatore-641006.

24/4/07
(Signature)
Mrs.V.Vanitha
**SUPERVISOR**
SeniorLecturer
Dept. of Computer Science &Engg.
Kumaraguru College of Technology.
Chinnavedampatti P.O.
Coimbatore-641006.

Submitted for Viva Voce Examination held on 25.04.07

25.4
Internal Examiner

25/4/07
External Examiner

P-1822

LEEMAC

A PROJECT REPORT

Submitted by

A.C.RAJ KUMAR 71203104029
R.VINOD 71203104057
M.V.DINESH KUMAR 71203104501

in partial fulfillment for award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2007

P-1822

ANNA UNIVERSITY: CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report "LATENCY AND ENERGY EFFICIENT MAC PROTOCOL" is the bonafide work of "A.C.Raj Kumar (71203104029), R.Vinod (71203104057) and M.V.Dinesh Kumar (71203104501)", who carried out the project work under my supervision.

(Signature)
Dr.S.Thangasamy

Head of the Department
Dept. of Computer Science & Engg.
Kumaraguru College of Technology.
Chinnavedampatti P.O.
Coimbatore-641006.

24/4/07
(Signature)
Mrs.V.Vanitha
SUPERVISOR
SeniorLecturer
Dept. of Computer Science &Engg.
Kumaraguru College of Technology.
Chinnavedampatti P.O.
Coimbatore-641006.

Submitted for Viva Voce Examination held on 25.04.07

25.4
Internal Examiner

25/4/07
External Examiner

P- 18221

# ACKNOWLEDGEMENT

We would like to extend our gratitude to the Principal of our college, **Dr.Joseph.V.Thanikal** M.E,Ph.D,PDF,EPIT for providing us the required resources to proceed with our project.

We would like to express our sincere thanks to **Dr.S.Thangasamy,** Dean, Department of Computer Science and Engineering, Kumaraguru College of Technology, for his guidelines that motivated us to develop a good product.

We are grateful to **Mrs.P.Devaki,** Assistant Professor, Department of Computer Science and Engineering, Kumaraguru College of Technology for her encouragement and support at various levels of this project work.

We also express our sincere thanks to **Ms.S.Rajini,** Senior Lecturer, Project Coordinator, Department of Computer Science and Engineering, Kumaraguru College of Technology for her valuable guidance and encouragement at every stage of this project.

We take immense pleasure in expressing our heartfelt thanks to our guide, **Mrs.V.Vanitha,** M.E, Senior Lecturer, Department of Computer Science and Engineering, Kumaraguru College of Technology for her inspiration and guidance throughout this project. We are very grateful to her for her help rendered to us in handling various tough spots during this project.

Most of all, we thank our parents and friends for their blessings, help and support without which we would not be able to do anything.

---

## ABSTRACT

---

# ABSTRACT

Numerous energy efficient protocols have been proposed for Wireless Sensor Networks. This project presents a proposal for the development of a energy aware Medium Access Control (MAC) protocol for wireless sensor networks that is appropriate for environmental monitoring purposes. This protocol saves energy by substantially reducing idle-listening. It is a reactive protocol which permits nodes to transmit packets every time they wake-up but at the same time proposes a mechanism to transmit packets that contain interesting data when nodes should not normally be awake.

The proposed protocol uses two techniques to reduce energy consumption for wide area large scale environmental monitoring applications. First, it uses a sleep/listen schedule in which nodes awake when a sample from the environment is taken. Second, it uses a mechanism through which a node, that has urgent packets to send, wakes up other nodes to convey data to base station for further processing. This project presents ns-2 simulation results. Results show that proposed protocol achieves less energy consumption and latency than the other protocols like S-MAC, D-MAC, etc.

---

## TABLE OF CONTENTS

## TABLE OF CONTENTS

CHAPTER NO TITLE PAGE NO

ACKNOWLEDGEMENT ii
ABSTRACT iii
LIST OF FIGURES
1. INTRODUCTION 1
  1.1 PROBLEM IDENTIFICATION 1
  1.2 JUSTIFICATION OF THE PROJECT 2
2. WIRELESS SENSOR NETWORKS 4
  2.1 INTRODUCTION 4
  2.2 GENERIC ARCHITECTURE 4
  2.3 FEATURES 6
3. MAC PROTOCOLS FOR WSNs 7
  3.1 DESIGN FACTORS 7
  3.2 PROBLEMS TO BE SOLVED 8
    3.2.1 ENERGY EFFICINCY 8
    3.2.2 SCALABILTY 10
    3.2.3 LATENCY, THROUGHPUT BANDWIDTH UTILIZATION, FAIRNESS 10
  3.3 OVERVIEW OF MAC PROTOCOLS 11
    3.3.1 S-MAC 11

      3.3.1.1 PERIODIC LISTEN AND SLEEP 11
      3.3.1.2 SYNCHRONIZATION 12
      3.3.1.3 CARRIER SENSE 22
      3.3.1.4 COLLISION AVOIDANCE 24
      3.3.1.5 OVERHEARING AVOIDANCE 25
      3.3.2.6 ADAPTIVE LISTENING 26
    3.3.2 T-MAC 29
    3.3.3 D-MAC 30
4. PROPOSED PROTOCOL 32
  4.1 PROTOCOL OPERATION 32
  4.2 MODIFIED SLEEP/LISTEN SCHEDULE 33
  4.3 TONE TIME SIGNALLING MECHANISM 35
5. ALGORITHMS 38
  5.1 SLEEP/LISTEN SCHEDULE 38
  5.2 TONE PACKETS MECHANISM 40
6. RESULTS 43
  6.1 LATENCY COMPARISON GRAPH 43
  6.2 ENERGY COMPARISON GRAPH 44
7. CONCLUSION 45
8. FUTURE ENHANCEMENTS 46
9. APPENDIX 47
  9.1 SAMPLE CODE 47
  9.2 SCREEN SHOTS 52
REFERENCES AND BIBLIOGRAPHY 56

## LIST OF FIGURES

## LIST OF FIGURES

FIG NO FIGURE NAME PAGE NO

2.2 ILLUSTRATION OF SENSOR NETWORK AND BACKBONE INFRASTRUCTURE 5
3.3.1 SENSOR FRAME FORMAT 11
3.3.1.1 PERIODIC LISTEN AND SLEEP 12
3.3.1.2 HIERARCHY OF PERIODS IN S-MAC 21
3.3.1.2 NEIGHBOR LIST UPDATING PERIOD SMALLER THAN SYNCHRONIZATION PERIOD 22
3.3.1.4 A SUCCESSFUL DATA TRANSMISSION BETWEEN TWO NODES 24
3.3.1.5 C AND D OVERHEAR THE TRANSMISSION BETWEEN A AND B 25
3.3.1.6 A THREE-HOP NETWORK WITH ONE SOURCE AND ONE SINK 27
3.3.1.6 TRANSMITTING A DATA PACKET THROUGH A THREE-HOP NETWORK,WITHOUT ADAPTIVE LISTENING 27
3.3.1.6 TRANSMITTING A DATA PACKET THROUGH A THREE-HOP NETWORK,WITH ADAPTIVE LISTENING 28
3.3 STAGGERED ACTIVE/SLEEP SCHEDULE

IN D-MAC 31
4.2 LISTEN TIME STRUCTURE 34
4.2 PACKET TRANSMISSION 35
4.3 TONE TIME SIGNALLING 36
5.1 VALUE SET BY NODE 39
5.1 SECOND SCHEDULE 40
5.2 TRIGGERED PACKET TRANSMISSION 41
5.2 MECHANISM TO SEND A TRIGGERED PACKET
BY SENDING TONE PACKETS 42
6.1 LATENCY COMPARISON GRAPH 43
6.2 ENERGY COMPARISON GRAPH 44
A.1 INITIAL STAGE BEFORE TRANSMISSION 52
A.2 BEGINNING OF DATA TRANSMISSION 53
A.3 MIDDLE STAGE OF DATA FLODDING 54
A.9 FINAL STAGE TRANSMISSION 5 55

---

# INTRODUCTION

---

# CHAPTER 1
# INTRODUCTION

## 1.1 PROBLEM IDENTIFICATION

The MAC is a sub layer of the Data Link Layer of the Open Systems Interconnection (OSI) model. Among other things, this layer is responsible for controlling the access of nodes to the medium to transmit or receive data. In sensor networks, all nodes cooperate for a single common task. At any particular time, one node may have dramatically more data to send than some other nodes. In this case fairness is not important as long as application-level performance is not degraded. In wireless sensor networks, MAC protocols control how sensor nodes access a shared radio channel to communicate with neighbours. Traditionally, this problem is known as the channel allocation or multiple access problem. Though MAC protocols have been extensively studied in traditional areas of wireless voice and data communications (e.g. Time division multiple access (TDMA), frequency division multiple access (FDMA), code division multiple access (CDMA), ALOHA and carrier sense multiple access (CSMA)), sensor networks require a MAC protocol that differs from those of traditional wireless voice or data networks in several ways. First of all, most nodes in sensor networks are likely to be battery powered and it is often very difficult to change batteries for all the nodes. Second, nodes are often deployed in an ad-hoc fashion rather than with careful pre-planning. Hence, after deployment the sensor nodes must quickly organize themselves into a communication network. Third, many applications employ large numbers of nodes. Finally, most traffic in the network is triggered by sensing events, and it can be

extremely bursty. All these characteristics suggest that traditional MAC protocols employed in the past wireless networks are not suitable for wireless sensor networks without modifications.

## 1.2 JUSTIFICATION OF THE PROJECT

The proposed energy-efficient MAC protocol is proposed to exploit the advantages that sampling data has from an energy saving perspective and, at the same time, cope with latency requirements of triggered data. Sampling data has two great advantages: first, the number of samples to take in a given period of time is known in advance and second, instants to take the samples are also known. This fact leads us to the idea that between two consecutive sample instants, the communication functions of two nodes is almost null. In this way, the proposed protocol exploits this fact to save energy turning off its radio between two consecutive sample instants for data transmissions. Significant energy savings can be achieved by this operation if we take into account that idle listening is the most energy consuming operation of all. However, if the radio is simply turned off, no triggered packets can be transmitted from originating nodes to the base station in a reasonable time. In such situation, triggered packets would be queued up and would also wait for the next available active time slot to be transmitted; what would create a long delay for triggered data, which would ideally have to be transmitted without delay. Furthermore, collisions would increase dramatically because all nodes in the network would content for the medium when the next time slot started. Proposed energy-efficient MAC protocol meets inherent requirements of triggered packets by creating a mechanism to wake-up nodes so they are able to move the packet through the network. It gives us the quick and better traffic load adaptation and also improves the latency by reducing the overhead.

Thus using this protocol we can save energy as well as improve the latency and the simulation results will justify this.

# WIRELESS SENSOR NETWORKS

# CHAPTER 2
# WIRELESS SENSOR NETWORKS

## 2.1 INTRODUCTION

Wireless sensor networks are dense networks of small, low-cost sensors, which collect and disseminate data. Wireless sensor networks facilitate monitoring and controlling of physical environments from remote locations. A sensor network is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it. It detects variations in temperature, sound, vibration, pressure, motion or pollutants. The sensor nodes self-organize their networks, rather than having a pre-programmed network topology.

Applications of wireless sensor networks include environmental control such as forest fire detection, air pollution and rainfall observation in agriculture; surveillance tasks of many kinds like intruder surveillance in premises; military monitoring and healthcare applications.

## 2.2 GENERIC ARCHITECTURE

Sensor networks are designed to transmit data from an array of sensors to a data repository or server that will store it for further analysis. Sensor nodes gather data autonomously, and the network passes this data to one or more base stations, which forward the data to a server. The wireless sensor networks which have sensing, computation and communication functions to move packets from sensor nodes to final servers, consume quantities of energy that must be taken into account to forecast the life cycle of a network and maximize it.

Sensor nodes are the network components that will be sensing and delivering the data. According to the system application requirements, nodes may do some computations. After computations, it transmits its data to its neighbouring nodes or simply passes the data as it is to the task manager. Sensor nodes can act as a source or sink in the sensor field. The function of a source is to sense and deliver the desired information. Thus, a source reports the state of the environment. On the other hand, a sink is a node that is interested in some information a sensor in the network might be able to deliver.

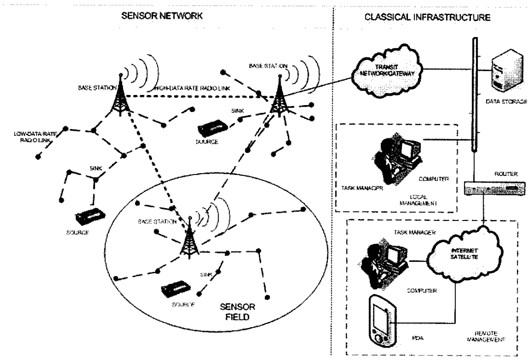


FIG 2.2 ILLUSTRATION OF SENSOR NETWORK AND BACKBONE INFRASTRUCTURE

## 2.3 FEATURES

The features for wireless sensor networks are listed as below

- Wireless sensor networks consist of many distributed sensor nodes. Each node has one or more sensors, an embedded processor, and a low-power radio, and is normally battery-operated.
- In some applications, sensor nodes are distributed within a vast expanse, such as earthquake monitoring in desert area. Sometimes their working environments are quite dangerous and even not accessible for humans.
- Normally, sensor nodes in one network collaborate for a common application. They communicate and exchange information in a peer-to-peer way (ad hoc fashion), instead of by accessing a base station.
- Sensor nodes keep silent for most of the time, but they will become active suddenly when something is detected.
- As time elapses in a network, some nodes may die, some nodes may move way, and some new nodes may be added. The topology may change over time.
- A message is a meaningful unit of data that a node can process. For saving energy, messages will be processed in a store-and-forward fashion, which is called in-network data processing, instead of being processed at a certain node in a centralized way.

**MAC PROTOCOLS FOR WSNs**

# CHAPTER 3
# MAC PROTOCOLS

Various protocols exist for Wireless Sensor networks. The main objective of these protocols is to reduce energy consumption. Some protocols concentrate on latency too. Apart from energy efficiency and latency the following factors are important as a guideline to design a protocol for Wireless Sensor networks.

## 3.1 DESIGN FACTORS

- **FAULT TOLERANCE**
  The failure of some sensor nodes must not affect the overall task of the network.
- **SCALABILITY**
  The protocol must be able to work in dense wireless sensor networks and also for expanding networks.
- **PRODUCTION COSTS**
  The cost of producing a node must be kept low.
- **HARDWARE CONSTRAINTS**
  A sensor node is made up of four basic components
    a sensing unit,
    a processing unit,
    a transceiver unit,
    a power unit.

  Apart from size, there are some other stringent constraints for sensor nodes. These nodes must consume extremely low power, operate in high volumetric densities, have low production cost, be dispensable and autonomous, operate unattended, and be adaptive to the environment.
- **SENSOR NETWORK TOPOLOGY**
  Wireless sensor network protocol must be able to maintain network topology so the network can accomplish its objectives.
- **POWER CONSUMPTION**
  In a multihop wireless sensor network, each node plays the dual role of data originator and data router. The malfunctioning of a few nodes can cause significant topological changes and might require rerouting of packets and reorganization of the network. Hence, power conservation and power management take on additional importance.

## 3.2 PROBLEMS TO BE SOLVED

There are some problems that are to be solved when designing a Wireless Sensor Network. These problems are discussed below,

### 3.2.1 ENERGY EFFICIENCY

Sensor nodes are battery-equipped. Due to their working environments, recharging or replacing batteries for each node is difficult and uneconomical, sometimes even impossible. Therefore, reducing the energy consumption to prolong the service lifetime of sensor nodes becomes a critical issue. This problem can be solved by attacking the sources of energy waste. The sources of energy waste are

- Idle listening
- Collision
- Overhearing
- Control packet overhead

- **IDLE LISTENING**

  For contention-based MAC protocols, such as IEEE 802.11 ad-hoc mode, in order to perform effective carrier sense against possible collisions, it puts nodes to listen to the channel all the time when nodes are idle. And radio will consume almost the same power as in receiving state. A considerable percentage of energy will be wasted on idle listening, especially when the traffic load on the network is light. Among those factors for energy waste, idle listening is a dominant one.

- **COLLISION**

  This may occur when neighbouring nodes contend for free medium and lossy channel will result in corruption of transmitted packets. When either of two cases happens, corrupted packets should be re-transmitted, which increases energy consumption.

- **OVERHEARING**

  This happens when a node receives some packets that are destined to other nodes.

- **CONTROL PACKET OVERHEAD**

  Exchanging control packets between sender and receiver also consumes some energy. The control packets include Request to Send (RTS), Clear to Send (CLS) and Acknowledgement (ACK).

### 3.2.2 SCALABILITY

For a wireless sensor network, its topology and size may change over time. So a good MAC protocol should accommodate to such changes.

### 3.2.3 LATENCY, THROUGHPUT, BANDWIDTH, UTILIZATION, FAIRNESS

These are common attributes for most of MAC protocols. The speed of changes on physical objects sensed by sensor nodes is much slower than the network speed. So latency is less important and can be tolerated in a certain range in such cases.

All MAC protocols are aimed at eliminating the above mentioned problems. Though it's not possible to eliminate all the problems, each protocol eliminates at least one problem.

## 3.3 OVERVIEW OF MAC PROTOCOLS FOR WSNS

### 3.3.1 S-MAC (Sensor MAC)

The features that S-MAC introduces are periodic sleep and listen. The basic idea is to let each node follow a periodic sleep and listen schedule. In listen period, the node wakes up for performing listening and communicating with other nodes. When sleep period comes, the nodes will try to sleep by turning off their radios. In this way, the time spent on idle listening can be significantly reduced, which accordingly saves a lot of energy, especially when traffic load is low. The duty cycle is defined as the ratio of listen period to a complete sleep and listen cycle. In S-MAC, the low-duty-cycle mode is the default operation for all nodes.

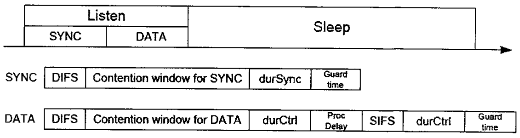


FIG 3.3.1 SENSOR FRAME FORMAT

#### 3.3.1.1 PERIODIC LISTEN AND SLEEP

The main technique used to reduce energy consumption in S-MAC is to make each node in the network follow a listen and sleep cycle. A complete cycle of listen and sleep period is called a *frame*. During sleep period, the node will turn off its radio if possible. In this way, a large amount of energy consumption caused by unnecessary idle listen can be avoided especially when traffic load is light. During listen period, the node may start sending or receiving packets if necessary. S-MAC provides a controllable parameter *duty cycle*, whose value is the ratio of the listen period to the frame length. In fact, the listen period is normally fixed according to some physical and MAC layer parameters. The user can adjust the *duty cycle* value from 1% to 100% to control the length of sleep period. Normally, the frame length is the same for all nodes in network.

The listen period is further divided into two parts. The first one called SYNC period is designed for SYNC packets, which are broadcast packets and used to solve synchronization problems between neighbouring nodes. The second one called DATA period is designed for transmitting DATA packets.

FIG 3.3.1.1 PERIODIC LISTEN AND SLEEP

#### 3.3.1.2 SYNCHRONIZATION

The Synchronization related components in S-MAC are described below:

**SYNC PACKET**

Each S-MAC node needs to exchange its schedule by periodically broadcasting a SYNC packet to its neighbours. The period of sending a SYNC packet is called the *synchronization period.* The default value in ns-2 is 10 frames (one frame = one sleep and listen cycle). Sending or receiving SYNC packets takes place in SYNC period.

### THE SYNC FRAME

| Field | Comment |
|---|---|
| type | Flag indicating this is a SYNC packet |
| length | Fixed size with 9 Bytes |
| srcAddr | ID of the sender |
| syncNode | ID of the sender's synchronization node |
| sleepTime | Sender's next sleep time from now |
| State | Indicating if the sender has changed its schedule recently |
| crc | Cyclic Redundancy Check |

The most valuable information in a SYNC packet is sleepTime, which tells all nodes that receive this packet when the next sleep period on the sender node comes. To avoid synchronization errors caused by clock drift on each node, the sleepTime uses a relative value rather than an absolute time.

### SCHEDULE TABLE

Each S-MAC node maintains one schedule table, which stores its own schedule and the schedules of all its known neighbours. There are two classes of schedules in the schedule table.

The node itself is on *primary schedule* and other schedules in the schedule table *secondary schedules.* A node may have no secondary schedule, for example, if all its neighbours follow the same schedule. But every node should have a primary schedule. The maximum number of records in the table is limited by a user-adjustable S-MAC parameter, which defines the maximum number of different schedules.

### FIELD DEFINITION OF SCHEDULE TABLE

| Field | Comment |
|---|---|
| txSync | Flag indicating need to send sync |
| txData | Flag indicating need to send data |
| numPeriods | Counter for sending sync periodically |
| numNodes | Number of nodes on this schedule |
| syncNode | The node who initialized this schedule |
| chkSched | Flag indicating need to check numNodes |

A schedule is actually a timer. Both primary schedule and other secondary schedules have their own timers. These timers run independently and their workings are quite similar.

### NEIGHBOUR LIST

Another important component in S-MAC is neighbour list. Each S-MAC node has to set up such a table to records the information of all its

known neighbours. The number of records in the list is also limited by a user-adjustable S-MAC parameter, which defines the maximum number of neighbours for each node. Like schedule table, neighbour list is established also through exchanging SYNC packets between neighbouring nodes.

### FIELD DEFINITION OF NEIGHBOUR LIST

| Field | Comment |
|---|---|
| nodeId | ID of this node |
| schedId | Schedule ID in schedule table that this node follows |
| active | Flag indicating this node is active recently |
| State | Flag indicating the node has changed schedule |

Neighbor list plays an important role in S-MAC. When S-MAC processes a unicast data request received from the upper layer, it will first check its neighbour list to see if the destination node is on the list. If not, the request will be refused. If yes, the flag txData of the schedule that the destination node follows is set (if no other sending request). Then when the next DATA period on this schedule arrives, the node will try to send out this packet.

### CHOOSING FIRST SCHEDULE

When a new node joins the network, it first listens for a fixed period (normally a synchronization period). We call this period *initial listening.* Either of the following two cases may happen.

1. No SYNC packet received during the initial listening.

   At end of this period, the node chooses a schedule by itself and sets a schedule timer for it. Normally the first cycle starts with SYNC period. Meanwhile the schedule is added to the first entry of its schedule table. To announce this new schedule, the flag txSync of this schedule is set, which indicates that the node will try to broadcast a SYNC packet in the next SYNC period.

2. Get a SYNC packet before the initial listening ends.

   That is the node's first SYNC packet received. It immediately follows the schedule that comes with the received SYNC packet, instead of choosing by itself after the initial listening. The value of sleepTime in the SYNC packet determines the starting point of the first cycle. Meanwhile the schedule is added to the first entry of the schedule table, and the sender of the SYNC packet is added to the neighbour list.

   The node also needs announce this schedule by setting the flag txSnyc in this schedule.

### UPDATING AND MAINTAINING SCHEDULES

After initial listening the node may get its first schedule by choosing itself or by following an existing one. These two cases are explained below.

1. A node receives the first SYNC packet.

   This may happen only after the node chooses the first schedule by itself. In this case, the node discards my its first schedule by removing it from its schedule table and follow the new one in the SYNC packet by adding it to the first entry of the schedule table. And the schedule timer that is associated with the discarded schedule will be rescheduled according to the value of sleepTime in the SYNC packet. The node that sent this SYNC packet will be added to its neighbour list as the first neighbour.

2. The node receives its first SYNC packet after it has chosen its first schedule.

The following five possible situations must be considered. For simplification, N represents the sender of the SYNC packet and S represents the schedule in the SYNC packet.

### GENERAL ALGORITHM FOR PROCESSING SYNC PACKETS

| S. No. | Condition | Action |
|---|---|---|
| 1 | N is a known neighbour in the node's neighbour list and it has not changed its primary schedule since receiving the last SYNC packet. | Update S in the node's schedule table by rescheduling its schedule timer with the value of sleepTime in the SYNC packet. This eliminates the clock drift between two nodes. |
| 2 | N is a known neighbour in the node's neighbour list, but it has switched its primary schedule to S, which is new to the node since it got its last SYNC packet. | Step1: *Process the schedule that N has switched from.*<br>The number of the nodes on this schedule has to be decreased by one. If the number goes to 0 after decrease, this schedule has to be removed from my schedule list.<br>Step2: *Process the new schedule S that N has switched to.*<br>If the node's schedule table is not full, S is added and assigned to a new schedule timer. Otherwise N has to be deleted from its neighbour list.<br>Step3: If the schedule that N switched from is my primary schedules check if that node is the only one on its primary schedule. If yes, the node chooses the next available schedule in its schedule table and sets it as its new primary schedule deleting the old one. |
| 3 | N is a known neighbour in my neighbour list, but it has switched its primary schedule to S, which is an existing one in the node's schedule table, since it got its last SYNC packet. | Step1: *Process the schedule that N switched from.*<br>The number of the nodes on the schedule that N used to follow has to be decreased by one. If the number goes to 0 after decrease, this schedule has to be removed from my schedule list.<br>Step2: *Process the schedule S that N has switched to.*<br>Update S in the node's schedule table by rescheduling its schedule timer with the value of sleepTime in the SYNC packet. And the number of nodes on S is increased by one. |
| 4 | N is an unknown neighbour to me and S is also new to the node. | If neither the node's schedule table nor neighbour list is full, S is added to its schedule table and a new schedule timer assigned to S. Then N is added to its neighbour list. |
| 5 | N is an unknown neighbour to the node but S is known in its schedule table. | Update S in the node's schedule table by rescheduling its schedule timer with the value of sleepTime in the SYNC packet. If the node's neighbour list is not full, N is added to it. And the number of nodes on S is increased by one. |

### PERIODICAL NEIGHBOUR DISCOVERY

In S-MAC, neighbouring nodes discover each other through exchanging SYNC packets. However, sometimes two neighbouring nodes may miss each other forever, for example, when they follow different schedules, whose SYNC periods do not overlap. Periodical neighbour discovery can prevent this from happening.

The basic idea of neighbour discovery in S-MAC is to make each node periodically execute neighbour discovery for a whole synchronization time. During neighbour discovery time, S-MAC will never try to go to sleep when its sleep period comes, so that the node can listen more time than usually and have more chances to hear a new neighbour.

The neighbour discovery period may vary, depending on the current number of its known neighbours.

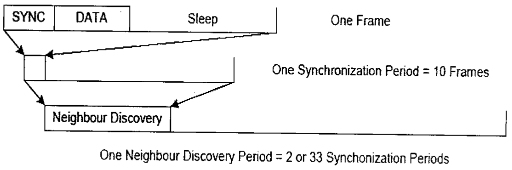


FIG 3.3.1.2 HIERARCHY OF PERIODS IN S-MAC

### PERIODICAL NEIGHBOUR LIST UPDATING

The schedule table and neighbour list will be updated each time when the node gets a SYNC packet from one of its neighbours. Except that, each S-MAC node also needs to check its neighbour list periodically to see if some of its neighbours have been inactive for a certain time (moved away or died for some reason), which need to be removed from the neighbour list.

Doing this is necessary and important, because it cannot only save space for the neighbour list, but also avoid unnecessary energy consumption caused by keeping trying to talk to an inactive neighbour, especially in a multi-hop mobile sensor network.

A timer is allocated to control periodical updating of the neighbour list. Its expiration period must be larger than the synchronization period.

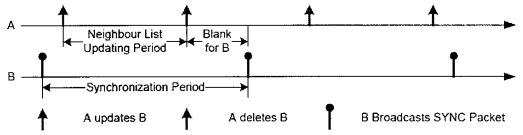


FIG 3.3.1.2 NEIGHBOUR LIST UPDATING PERIOD SMALLER THAN SYNCHRONIZATION PERIOD

### 3.3.1.3 CARRIER SENSE

Carrier sense in S-MAC is performed both through physical and virtual mechanisms. The medium is determined as free only when both of them indicate that it is free.

Physical carrier sense is performed at physical layer by checking the current radio state. Every time when the radio starts receiving or transmitting, the PHY layer will inform the MAC thereof. It happens also when the receiving or transmitting is over.

Obviously, the medium will be determined as busy when radio is in receiving or transmitting state.

S-MAC defines two Network Allocation Vector (NAVs) for virtual carrier sense. One is simply called NAV and used to indicate if the medium is busy or not. The other one is called Neighbor NAV, which tracks its neighbours' NAV.

Only when both of the NAVs have counted down to zero, the virtual carrier sense indicates the medium is free. The structure of and the way of running for two NAVs are quite similar.

**DIFFERENCES BETWEEN NAV AND NEIGHBOUR NAV**

| NAV | Neighbor NAV |
|---|---|
| Receive RTS destined to other node | Receive RTS destined to me |
| Receive CTS destined to other node | After CTS is sent out (for DATA timeout) |
| Receive DATA destined to other node | Receive DATA destined to me |
| Receive ACK destined to other node | After DATA (unicast) is sent out |
| Error found in received packet, NAV is updated by an EIFS value | After ACK is sent out |
| Collision detected when the radio is receiving a packet and another packet arrives, NAV will be updated by an EIFS value right after the radio finishes receiving | ACK timeout, but not reached maximum number of times to extend Tx time |

## ROLE OF NAVS

1. Neighbour NAV will act as a timer for DATA timeout on the node, which has sent out the CTS packet and is waiting for the arrival of the DATA packet.
2. Adaptive listening will be triggered when either of the two NAVs counts down to zero.

### 3.3.1.4 COLLISION AVOIDANCE

RTS (Request to Send) /CTS (Clear to Send) mechanism can efficiently reduce the durations of collisions.

Prior to the actual DATA packet, the sender should exchange RTS and CTS packets with the receiver. But only unicast packets follow the sequence of RTS/CTS/DATA/ACK. Broadcast packets will be sent without using RTS/CTS.

All immediate neighbours of both the sender and the receiver will learn the coming transmission from the RTS or CTS packets and will keep silent during the reserved period. Although the combination of carrier sense and RTS/CTS mechanisms can largely reduce the probability and durations of collisions, collisions cannot be completely avoided. If two neighbouring nodes finish carrier sense and start sending at almost the same time, collision will occur.

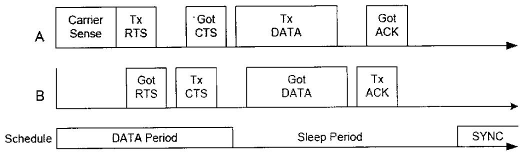


FIG 3.3.1.3 A SUCCESSFUL DATA TRANSMISSION BETWEEN TWO NODES

### 3.3.1.5 OVERHEARING AVOIDANCE

Overhearing is one of the major sources of energy waste. Overhearing takes place on a node when it receives some packets that are destined for other nodes. In 802.11, measures like latency and bandwidth utilization are considered in the first place. To achieve better performance in a shared-medium network, carrier sense, especially virtual carrier sense should be performed more efficiently. The best way to achieve it is to let each node keep listening to all its neighbours' transmissions.

This method will lead to large amounts of energy consumptions on overhearing, especially when node density is high and the traffic load in the network is heavy.

For S-MAC, saving energy is its primary goal. To avoid overhearing, S-MAC forces interfering nodes to go to sleep after they receive an RTS or a CTS packet that is not destined for them. In this way, nodes that interfere their neighbours' transmissions will not hear DATA packets, which normally take much longer transmission time than control packets, and following ACK packets.

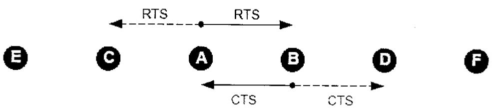


FIG 3.3.1.5 C AND D OVERHEAR THE TRANSMISSION BETWEEN A AND B

The above figure shows a five-hop linear network. Each node can only hear its immediate neighbours. We assume that all the nodes share the same schedule. Suppose A is communicating with B lets first see which nodes should go to sleep during this transmission.

Collision only happens on the receiver side. D is supposed to go to sleep, because its transmission interferes with B's reception of the DATA packet. C is two hops away from B, so C's transmission will not interfere with B's reception. But if C talks to E while A is sending data to B, C will not receive any packet from E because collision happens on C. C's transmission is a waste of energy and it also needs to go to sleep. Both E and F are at least two hops away from the nodes that are transmitting, and they will never produce interference. Therefore, E and F have no need to sleep. Those nodes that are the immediate neighbours of the sender or the receiver should go to sleep.

### 3.3.1.6 ADAPTIVE LISTENING

Adaptive listening is the dominant technique that S-MAC has proposed to efficiently and largely reduces the latency in a multi-hop transmission.

Let us look at an example to see how S-MAC transmits a packet in a three-hop linear network. Suppose there are four nodes A, B, C, and D, which are put in a line, each node can hear only its immediate neighbours. A is the source and D is the sink. Now a data packet is generated at the source node and destined to the sink node. We assume all nodes share the same schedule and no message passing technique is applied.

FIG 3.3.1.6 A THREE-HOP NETWORK WITH ONE SOURCE AND ONE SINK

When adaptive listening is not employed, each node has at most one chance to send the data packet out in each frame, because checking to send the data packet only takes place at the beginning of each DATA period. If the arrival of the data packet at a certain node misses this checkpoint of this frame, it has to wait until the next DATA period comes. That is to say, when each node strictly follows its sleep schedule, there is a potential delay on each hop. We can see from the Fig. 2.11 that we need three frames to transmit the data packet from the source A to the sink D. The data packet travels only one hop in each frame.

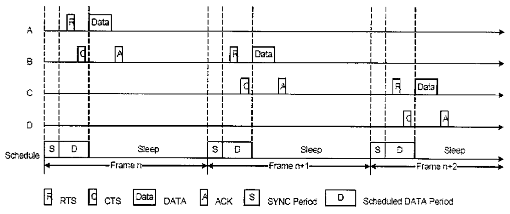


FIG 3.3.1.6 TRANSMITTING A DATA PACKET THROUGH A THREE-HOP NETWORK, WITHOUT ADAPTIVE LISTENING

Now we consider the case if adaptive listening is applied. The basic idea is that at the end of one transmission, S-MAC will give those nodes, which are involved in this transmission including sender, receiver, and neighbours of both them that overhear this transmission, another chance of transmitting data packets. In this way, one node may have two DATA periods for sending or receiving data packets in a frame.

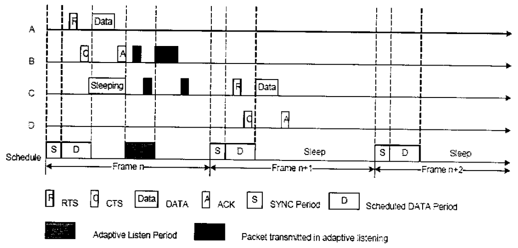


FIG 3.3.1.6 TRANSMITTING A DATA PACKET THROUGH A THREE-HOP NETWORK, WITH ADAPTIVE LISTENING

As shown in the diagram, transmitting the data packet from A to B (including carrier sense and exchange of RTS/CTS) starts from the scheduled DATA period and extends to the following sleep period. Both the sender A and the receiver B will use their Neighbor NAVs to perform virtual carrier sense, while C is overhearing this transmission and will use its NAV for virtual carrier sense. Both NAV and Neighbor NAV contain the same value, which is the time reserved for the current transmission. The expiration event of those NAVs (count down to zero, indicating the end of the current transmission) will trigger the execution of adaptive listening on their owner nodes.

## 3.3.2 T-MAC

Timeout-MAC (T-MAC) is proposed to enhance the performance results of S-MAC protocol under variable traffic load. In T-MAC, the listen period ends when no activation event has occurred for a time threshold (TA). This operation makes T-MAC's schedule variable instead of the fixed schedule proposed in S-MAC. T-MAC synchronization is similar to that of S-MAC. SYNC packets are exchanged between nodes to form virtual clusters that share the same synchronization. A node can run more than one synchronization scheme. The scheme used to contend for the medium is the well known RTS/CTS/DATA/ACK. However, T-MAC proposes a change in this scheme that is used to avoid the early sleep problem. The early sleep problem is the excessive contention for a node that wants to transmit to its neighbours. To avoid this problem T-MAC proposes two solutions as follows. Future request is to send and priority on full buffers. Simulations have shown that the T-MAC protocol introduces a way of decreasing energy consumption in a volatile environment where the message rate fluctuates, either in time or in location.

Implementation of the T-MAC protocol has shown that, during a high load, nodes communicate without sleeping, but during a very low load, nodes will use their radios for as little as 2.5% of the time, saving as much as 96% of the energy compared to a traditional protocol.

### 3.3.3 D-MAC

The main objective of DMAC is to achieve very low latency and still be energy efficient. For that purpose it is designed to overcome S-MAC problems: increased delivery latency because a packet can not reach the sink in a single listen period, fixed duty cycles that do not adapt to traffic changes and the increased possibility of collisions due to synchronous duty cycle. A problem that exists in implicit *sleep delay* reducing protocols: SMAC and T-

MAC. *Sleep delay* refers to the time that a packet suffers since it is transmitted from originating node to the sink (Base Station). This delay refers to the time that transmitting node has to wait for intended receiving node to wake up and receive the packet. S-MAC and T-MAC only solve this problem for a two-hop path. If a network is a multi-hop one with more than two hops, the solutions provided by S-MAC and TMAC are not appropriate. This problem is identified as *Data forwarding Interruption.* The limited overhearing range due to radio sensitivity is the origin of this DFI problem. Low latency is achieved by assigning subsequent slots to nodes that are successive in the data transmission path. With this scheme it is expected that a packets do not suffer from sleep delay at all because the next intended receiving node must always be awake when transmitting node wants to transmit a packet to it.

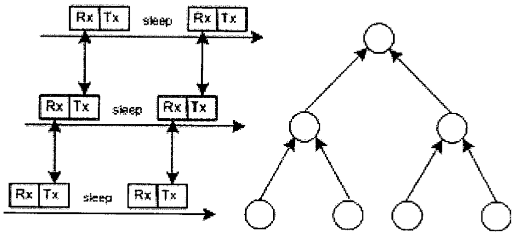


FIG 3.3.3 STAGGERED ACTIVE/SLEEP SCHEDULE IN D-MAC

DMAC staggers the active/sleep schedule of the nodes in the data gathering tree according to its depth in the tree. This allows continuous packet forwarding flow in which all nodes on the multi-hop path can be notified of the data delivery in progress as well as any duty-cycle adjustments. Simulation results have shown that DMAC achieves both energy savings and low latency when used with data gathering trees in wireless sensor networks.

## PROPOSED PROTOCOL

# CHAPTER 4

# PROPOSED PROTOCOL

## 4.1 PROTOCOL OPERATION

There are two types of data in wireless sensor networks: sampling and triggered. *Sampling* data is obtained by sampling a certain parameter a given number of times every day while *triggered* data is disseminated after a certain event has happened. For energy saving purposes, it is important to differentiate between these two types of data.

The proposed energy-efficient MAC protocol is proposed to exploit the advantages that sampling data has from an energy saving perspective and, at the same time, cope with latency requirements of triggered data.

Sampling data has two great advantages: first, the number of samples to take in a given period of time is known in advance and second, instants to take the samples are also known. The proposed protocol saves energy turning off its radio between two consecutive sample instants for data transmissions. Significant energy savings can be achieved by this operation if we take into account that idle listening is the most energy consuming operation of all. However, if the radio is simply turned off, no triggered packets can be transmitted from originating nodes to the base station in a reasonable time. In such situation, triggered packets would be queued up and would also wait for the next available active time slot to be transmitted; what would create a long delay for triggered data, which would ideally have to be transmitted without delay. Furthermore, collisions would increase dramatically because all nodes in the network would content for the medium when the next time slot started.

Proposed energy-efficient MAC protocol meets inherent requirements of triggered packets by creating a mechanism to wake-up nodes so they are able to move the packet through the network. Specifically, it is proposed a third time slot inside the listen interval within a frame. This time slot must be large enough to permit nodes to transmit and receive a very small packet and much smaller than data slot times. This time slot will be useful to signal other nodes to wake up at instants they are supposed to be sleeping to avoid idle listening.

The proposed energy-efficient MAC protocol inherits some characteristics of S-MAC with coordinated adaptive sleeping. The RTS/CTS/ACK mechanism is used for exchange of packets between nodes.

## 4.2 MODIFIED SLEEP/LISTEN SCHEDULE

This protocol makes some additions to sleep/listen schedule in S MAC to meet the requirements of triggered packets and at the same time avoid idle listening. SYNC interval is a time slot used to send synchronization information and DATA is used for RTS/CTS/DATA sequence. In the proposed protocol, between two sets of active listen periods the radio is turned off since the very beginning of the SYNC interval. Therefore, there is no way to signal other nodes to wake up is a sudden event occurs.

The proposed energy efficient MAC protocol introduces a new time interval that will be used for a node that has detected an extraordinary event and generated its correspondent packets to signal other nodes in the wireless sensor network that there are triggered packets that need to be disseminated through the network as soon as possible. With this scheme, triggered packets

will be sent in the next active listen period of the underlying S-MAC synchronization.

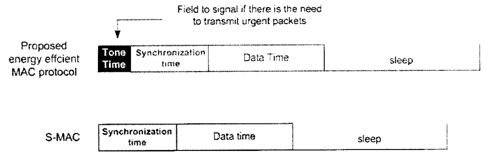


FIG 4.2 LISTEN TIME STRUCTURE

The proposed energy efficient MAC protocol novel listen-time structure will overcome initially proposed sleep/listen schedule drawbacks in the following ways:

- **DECREASED LATENCY**

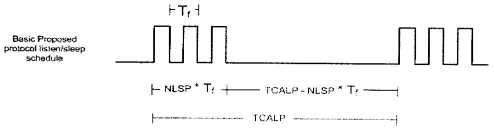
  Triggered packets will not wait a time between a Time Frame (Tf) and TCALP-NLSP*Tf

- **DECREASED QUEUE LENGTH**

  Since triggered packets will be sent in the next active listen period of the underlying S-MAC synchronization, nodes will not queue up triggered packets for more than a Tf and therefore queue length is lowered.

- **LESS NUMBER OF COLLISIONS**

  The number of packets to transmit are much less than if they were accumulated in the TCALP-NLSP*Tf time. Thus, the probability of collisions decreases dramatically.



Where:

$T_f$: Time Frame

NLSP : Number of Sleep/Listen Periods within a set

TCALP: Time between Consecutive sets of Active Listen Periods

TCALP - NLSP * $T_f$: Maximum time a triggered packet has to wait to be transmitted in basic schedule

FIG 4.2 PACKET TRANSMISSION

## 4.3 TONE TIME SIGNALING MECHANISM

The Tone Time interval will be used by nodes which have sensed an interesting event and thus generated triggered packets that need to reach the Task *Manager* for urgent analysis.

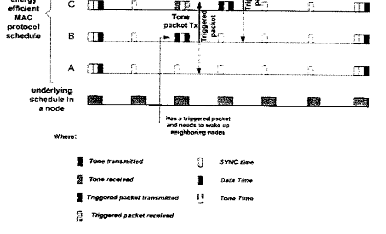


FIG 4.3 TONE TIME SIGNALLING

Latency and Energy efficient MAC protocol listen/sleep schedule optimized to avoid idle listening and decrease latency, collisions and queue length for triggered packets.

The proposed sleep/listen mechanism is explained in the following paragraphs:

- The proposed energy efficient protocol sleep/listen schedule is based on the underlying schedule that resulted from initial synchronization between nodes. This underlying synchronization is based on the most simple sleep/listen schedule of SMAC. The important parameters here are *Duty Cycle*, *Sync Time duration* and *Data Time Duration*.
- Two important parameters TCALP and NLSP are introduced.
- Within the TCALP-NLSP*Tf, radio is turned off in Data and SYNC times. However, all nodes turn on its radio at Tone Time to listen for possible requests from other nodes, to wake up and transmit triggered packets. Since all nodes have its radio turned on in Tone Time, some energy is consumed in idle listening. Therefore, it is crucial to design the Tone Time as small as possible compared to SYNC and DATA times; but large enough to allow a node to transmit/receive a very small Tone-packet.
- Whenever a interesting event is sensed, e.g. a sudden raise in temperature or another monitored phenomenon, the node generates a corresponding triggered packet that must reach the Task Manager with minimum latency.
- Once a node has generated a triggered packet it broadcasts a Tone-packet indicating that it needs the next-hop node to wake up so the packet can be disseminated though the network until it reaches the Base Station.

- Neighboring nodes receive Tone-packet and check its destination address. If the packet is intended for it, they do not go to sleep when the Tone Time is over.
- The node that originated the Tone packet sends the triggered packet. After sending the packet it goes to sleep following the original schedule to accommodate the sampled packets.
- The process is repeated as many times as number of nodes are in packet's route to the nearest Base Station.

---

## ALGORITHMS

---

# CHAPTER 5
# ALGORITHMS

The algorithms of the proposed protocol are explained here.

## 5.1 SLEEP/LISTEN SCHEDULE

The original sleep/listen schedule algorithm is modified and the new sleep/listen schedule works as described below:

- When a node is created it sets its parameters to:
- • *Status = 1*. This means that the node will follow at the beginning the original S-MAC schedule.
- • *NLSP = 0*. Since the node is following the S-MAC schedule at the beginning, it does not need to set NLSP to any value because the node will wake up periodically anyway
- *Status* is a variable that will control the number of periods a node stays slept once the node follows the proposed protocol schedule. Thus, every time a new cycle (Tone->Sync->Data.>Sleep) begins with a Tone Time, this variable's value must be decreased by one.
- A node stops to follow the S-MAC protocol schedule and starts to follow the proposed protocol one when it receives the first ACK packet from its neighbour indicating that it successfully received the data packet. After receiving the first ACK packet the node sets its parameters to:
    - o *Status = X*. Status is set to a specific value so the node will not turn its radio on in the next period because the sampling packet

has already been sent (due to ACK packet received). Status variable is decreased in each iteration.

  - *NLSP = Y*. Independently of the value of Status variable, NLSP will make the node turn its radio on when different than zero. Every time an ACK packet is received, the node sets these values again. Status and NLSP are the variables that determine the configuration of the new schedule.

- If *Status* = 0 then that node is either following the S-MAC schedule or ending its configured number of periods to sleep, so it must turn on its radio in the Tone Time. Otherwise, the node must sleep to satisfy the number of periods to sleep set on the *Status* variable.
- If NSLP is different than zero then that node has already synchronized with an ACK packet and must turn its radio on for NLSP number of periods before it goes to sleep In each iteration NLSP is decreased.
- After a node 'decides' to turn on its radio on or not, it must follow the natural cycle: Sync->Data.>Sleep->Tone.

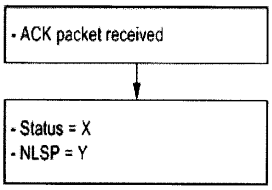


FIG 5.1 VALUE SET BY NODE

When An ACK packet is received the node sets new values.

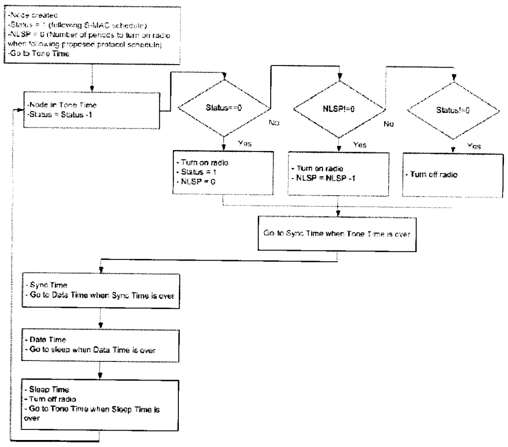


FIG 5.1 SECOND SCHEDULE

Proposed protocol scheme to create and use a second schedule in top of existing one.

### 5.2 TONE PACKETS MECHANISM

The algorithm to implement the sleep/listen schedule works as described below:

- When a node is created there is no need to send Tone packets since nothing has been sensed yet.
- A node follows its normal cycle (Tone->Sync->Data->Sleep) when there is no need to transmit Tone packets.
- If an interesting event is sensed at any time, a corresponding triggered packet is generated and then, the node knows it must send Tone packets.
- If there is a need to transmit a Tone packet, the node turns on its radio no matter the values of variables *Status* or *NLSP* described in previous subsection. It contents for the medium and after it has gained the medium, it broadcasts the packet.
- After broadcasting Tone packets the node supposes the other nodes received it and than is ready to send its triggered packets.
- Triggered packets are sent in Data time.

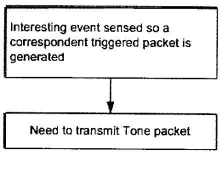


FIG 5.2 TRIGGERED PACKET TRANSMISSION

A need to transmit a Tone packet is generated when an interesting event is sensed.

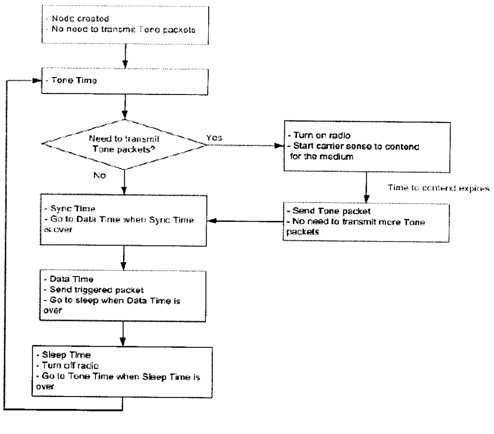


FIG 5.2 MECHANISM TO SEND A TRIGGERED PACKET BY SENDING TONE PACKETS.

## RESULTS

---

# CHAPTER 6

# RESULTS

## 6.1 LATENCY COMPARISON GRAPH

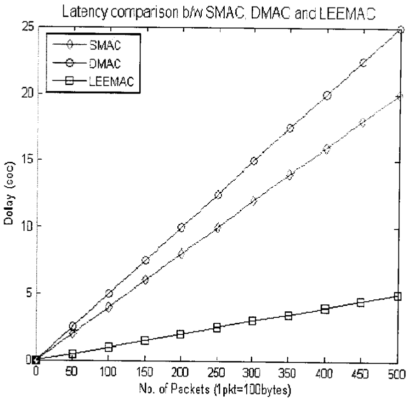


FIG 6.1

---

## 6.2 ENERGY COMPARISON GRAPH

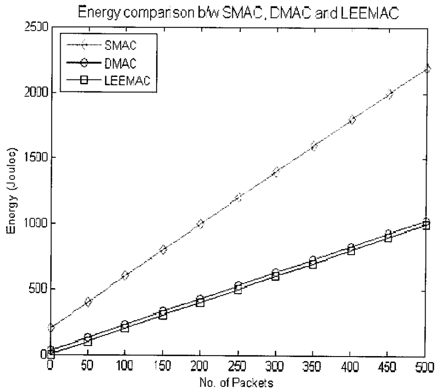


FIG 6.2

---

## CONCLUSION

# CHAPTER 7
# CONCLUSION

We proposed an energy aware MAC protocol for wireless sensor networks that gathers data for wide-area large scale environmental monitoring. The scheme saves energy by organizing the networks usage changing the running synchronization. Specifically, the
Proposed protocol uses:

- A sleep/listen schedule running in top of a previously negotiated one. In this schedule, nodes wake up only when a sample is to be taken. This schedule is intended to save more energy by avoiding idle listening.
- A mechanism to wake-up nodes when a node has the urgency to transmit a triggered packet. We called this mechanism Tone-Time; which met the requirements of triggered packets of low latency.
- According to simulation results, the proposed scheme is observed to perform better in terms of achievable network lifetime as compared to similar existing schemes like SMAC.
- Furthermore, simulations show that:
- The higher the rate of transmission of packets (sampling packets) the less the consumption of energy.
- At a fixed Duty Cycle (DC), the higher the message inter-arrival period the greater the difference of consumed energy between Proposed Protocol and SMAC.

## FUTURE ENHANCEMENTS

# CHAPTER 8
# FUTURE ENHANCEMENTS

In the near future, the present short comings are to be overtaken and the proposed protocol is aimed to be implemented in a sensor network platform and its performance is evaluated through real experiments. Then we will add more features to the sleep/listen schedule in order to make it more efficient.

## APPENDIX

# CHAPTER 9

# APPENDIX

## 9.1 SAMPLE CODE

**Neya.tcl**

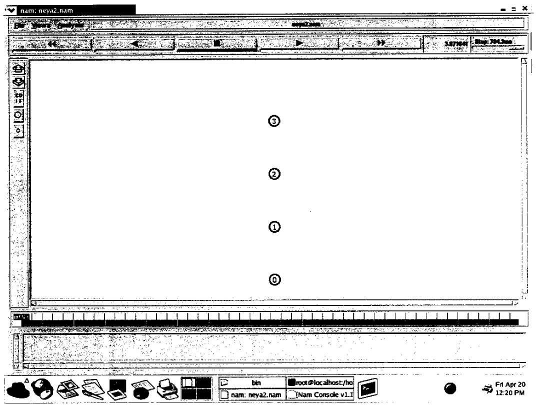
```
set val(x)          600    ;# X dimension of the topography
set val(y)          600    ;# Y dimension of the topography
set val(cp)         "../mobility/scene/cbr-50-10-4-512"
set val(sc)         "../mobility/scene/scen-670x670-50-600-20-0"
set val(ifqlen)            500          ;# max packet in ifq
set val(nn)         4            ;# number of nodes
set val(seed)       0.0
set val(stop)       10000.0;#65.0              ;# simulation time
set val(tr)         neya1.tr     ;# trace file
set val(nam)        neya1.nam  ;# animation file
set val(lm)         "off"       ;# log movement
set val(energymodel)   EnergyModel     ;
set val(radiomodel)        RadioModel     ;
set val(initialenergy)  10000          ;# Initial energy in Joules


Mac/SMAC set syncFlag_ 1
Mac/SMAC set dutyCycle_ 10
set ns        [new Simulator]
set tracefd   [open neya2.tr w]
```

```
set windowVsTime2 [open neya2.tr w]
set namtrace    [open neya2.nam w]

$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)

set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)


#
# Create god
#
create-god $val(nn)

#global node setting
     $ns node-config -adhocRouting DumbAgent \
                    -llType $val(ll) \
                    -macType $val(mac) \
                    -ifqType $val(ifq) \
                    -ifqLen $val(ifqlen) \
                    -antType $val(ant) \
                    -propType $val(prop) \
                    -phyType $val(netif) \
                    -channelType $val(chan) \
                    -topoInstance $topo \
                    -agentTrace ON \
                    -routerTrace ON \
```

```
                    -macTrace ON \
                    -energyModel $val(energymodel) \
                    -idlePower 1.0 \
                    -rxPower 1.0 \
                    -txPower 1.0 \
              -sleepPower 0.001 \
              -transitionPower 0.2 \
              -transitionTime 0.005 \
                    -initialEnergy $val(initialenergy)
$ns set WirelessNewTrace_ ON
      for {set i 0} {$i < $val(nn) } {incr i} {
            set node_($i) [$ns node]
      }
#initial positions to node
$node_(0) set X_ 300.0
$node_(0) set Y_ 10.0
$node_(0) set Z_ 0.0
$node_(1) set X_ 300.0
$node_(1) set Y_ 160.0
$node_(1) set Z_ 0.0
$node_(2) set X_ 300.0
$node_(2) set Y_ 310.0
$node_(2) set Z_ 0.0
$node_(3) set X_ 300.0
$node_(3) set Y_ 460.0
$node_(3) set Z_ 0.0
set tcp [new Agent/UDP]
```

```
$ns at $val(stop) "stop"
proc stop {} {
      global ns tracefd namtrace
      $ns flush-trace
      close $tracefd
      close $namtrace
      }
$ns run
```

## 9.2 SCREEN SHOTS



A.1 INTIAL STAGE BEFORE TRANSMISSION

## REFERENCES AND BIBLIOGRAPHY

### PRINT MEDIA

1. Archana Bharathidasan, Vijay Anand Sai Ponduru, “Sensor Networks: An Overview”
2. V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, “Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor
3. W. Ye, J. Heidemann, and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks”
4. Y.-B. Ko, V. Shankarkumar, and N. H. Vadiya, “Medium access control protocols using directional antennas in ad-hoc networks,"

### ONLINE SOURCES

1. http://www.isi.edu/nsnam/ns/tutorial/index.html
2. http://www.isi.edu/scadds/projects/smac/
3. http://www.isi.edu/nsnam/ns/
4. http://www.sensornetworks.edu/ns/smac
5. http://www.wirelessnetworks.edu