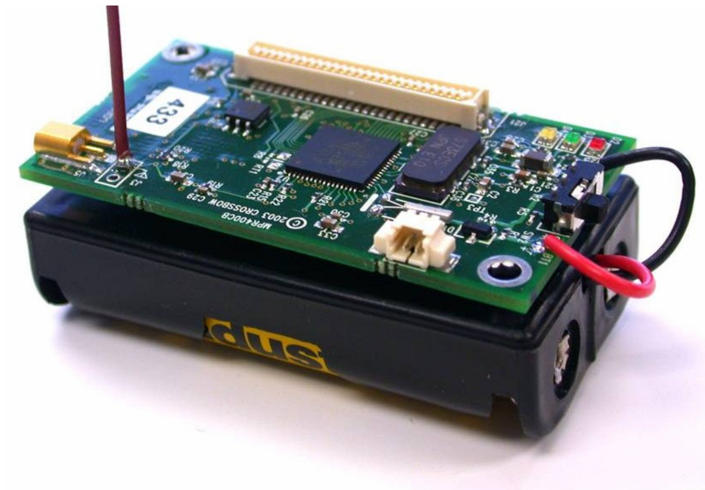


Polling as MAC protocol in a Multi-hop Sensor Network

CSC575 – Project Report

Prasanna Jeevan Devanur Nagabhushan (pdevanu)
Indira Radhika Pulla (irpulla)
Varun Tutpet Keshava Murthy (vtutpet)
Sandeep Budanur Ramanna (sbudanu)

Table of Contents

1. Overview

We have implemented "Multi-hop Polling"as a MAC protocol for our project.

A simple "Polling mechanism" is a centralized scheme with one Base-Station and several slave stations. The base-station polls all the mobile stations in a round-robin fashion. The master does establish the list of stations wishing to transmit during the contention phase. After this phase, the base-station polls each station on the list. After all the stations have been polled, the base-station polls again. At any point time, only one station can transmit thus avoiding collision of data packets. Also the QoS guarantees can be made such as delay, delay jitter and bandwidth to applications that need these QoS characteristics.

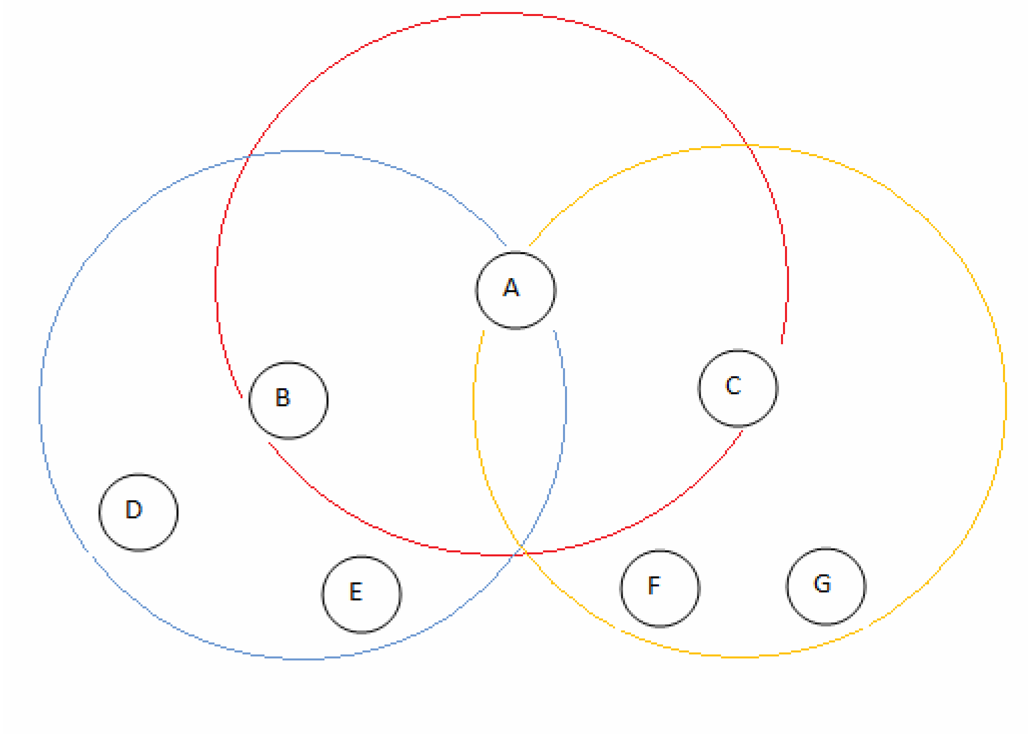


Figure-1: Multi-hop polling network topology

In a multi-hop network, all the nodes wishing to transmit might not be in the same range of the Base station. Hence, a hierarchy is built, in which the intermediate nodes in higher level will act as 'Base Station' to the nodes in the lower level.

In the figure-1, the node A acts as a base station for B and C and the nodes B and C will "act" as base station for nodes D, E and F, G respectively.

The multi hop hierarchy can be represented in a tree structure, which is shown in the figure-2. Here the base-station is at level-0, mobile stations B & C are at level-1 and D,E,F & G are at level-2.

2. Introduction

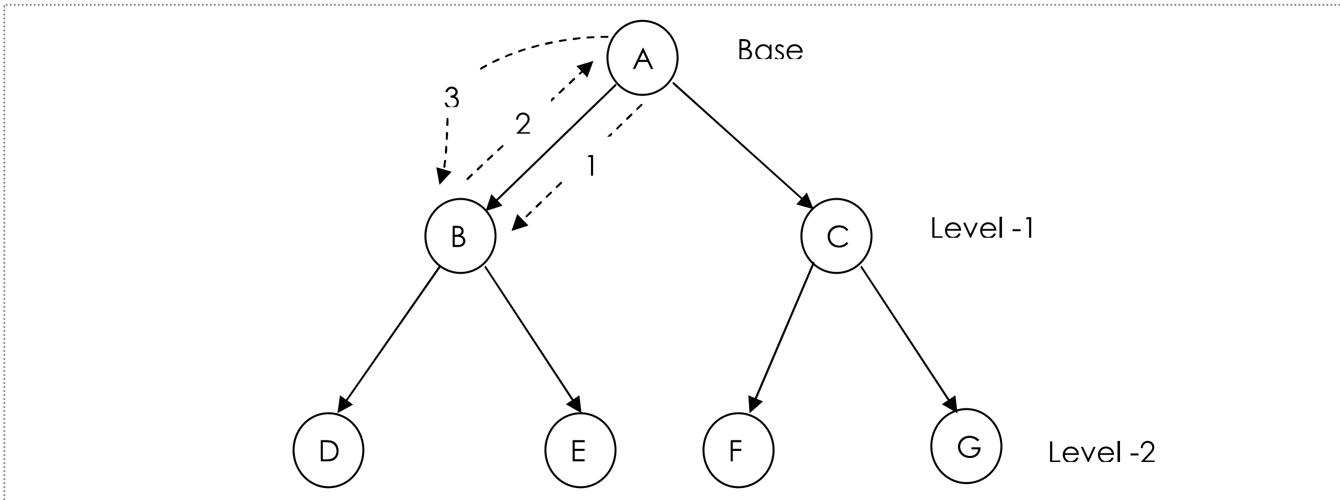


Figure-2: Tree representation of the multi-hop polling network

----- 1 -------▸ Broadcast Control

----- 2 -------▸ Join Request

----- 3 -------▸ Polling Message

In this MAC protocol each Base Station (BS) sends out a control message at regular intervals communicating that it is the base station. Any Mobile Station (MS) which wishes to join the network responds to the 'BS Broadcast Control Message' with a 'Join Request Message'. Since all the MS's within the range of the BS can respond to the control message at the same time, collisions might occur. To reduce the chances of collision, each child station waits for a random time interval before sending the 'Join Request Message'. After the BS accepts the Join Request message, the MS will be added in the polling cycle of the BS. MS includes its address (mote ID) in the Join Request message. The base station polls each MS in a sequential manner using the mote ID as destination address. The reservation in one polling cycle is carried over to the next polling cycle implicitly unless child timeouts occur.

Since we are implementing 'Polling in a Multi hop Wireless network', each MS in the network can in turn act as Base Station to its child nodes. These nodes are called the "Acting Base Stations" (ABS). So, an Acting Base station in turn does the following actions in a recursive manner for its child nodes.

1) Send out a Broadcast Control Message
2) Receive the Join request Messages to add child nodes to the polling list
3) Transmit it's own data
4) Poll the child nodes in sequence
5) Allow child nodes to transmit data
6) Send Polling response to its parent node after the polling cycle is completed

The BS includes guard space before polling the next MS.

Note: A node which already has a parent Base-Station will discard subsequent BS control messages from other nodes.

3. Messages

The various messages used and its codes are as follows

Control Code	Message	sub code	Description
1	Base-station broadcast message	-	Base Station sends this message to alert nearby child nodes for registration.
2	Join Request	-	Child sends to a base station to register for polling.
3	Polling	-	Base station sends this message to inform a child node of its data transmission slot.
4	Data Message	-	Base station/Child nodes send this message to transmit user data.
5	Polling Response	-	Child node(or ABS) sends this message to its Base station to indicate the completion of its data transmission and polling activity.

Table 1: Message Types

The description of each message is as follows:

1. **Base-station control message (BCM):** This is a broadcast message sent from the base-station (actual or acting) at the start of every Polling cycle. In this message the base station sends its station id and mentions that it is the base station. Base-station control message contains two fields, one is control code and the other contains its address (moteID).

2. **Join Request (JREQ):** This is a unicast message from a child station to the base station requesting the base station to add it in the polling cycle. This message is sent in response to the broadcast message.

3. **Polling:** This is a unicast message from a base station to the child station indicating the start of child's data transmission slot. Once a child station receives this message, it can transmit its own data and poll its own child stations.

4. **Polling Response Message:** Child station (or ABS) sends this message to its Base station to indicate the completion of its data transmission and polling activity. Base station on receiving this message jumps to polling next station in the list.

5. **Data Message:** This is a Unicast message used for the actual data transmission by both base stations and child stations (or ABS).

4. Message Format

DESTINATION ADDRESS		SOURCE ADDRESS	
TYPE	GROUP	LENGTH	DATA CRC
DATA			

Figure-3: Structure for Polling MAC Messages

Tiny OS message structure is modified to accommodate polling MAC requirements.

5. Timing Diagram

The figure-4 represents the timing diagram of the stations at each level. Here, we consider the Base-station(BS) is at level-0, the child stations directly connected to BS will be at level-1, the child stations which have 'Acting Base Station' as their parent will be at level-2 and so on.

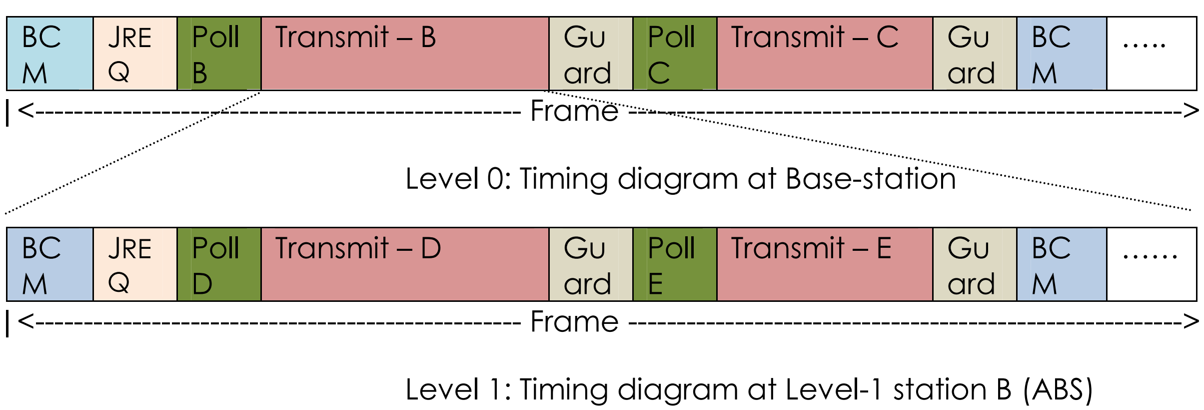


Figure-4: Timing Diagram at various levels

6. State Transition Diagram

6.1. State Transition Diagram for Base-Station

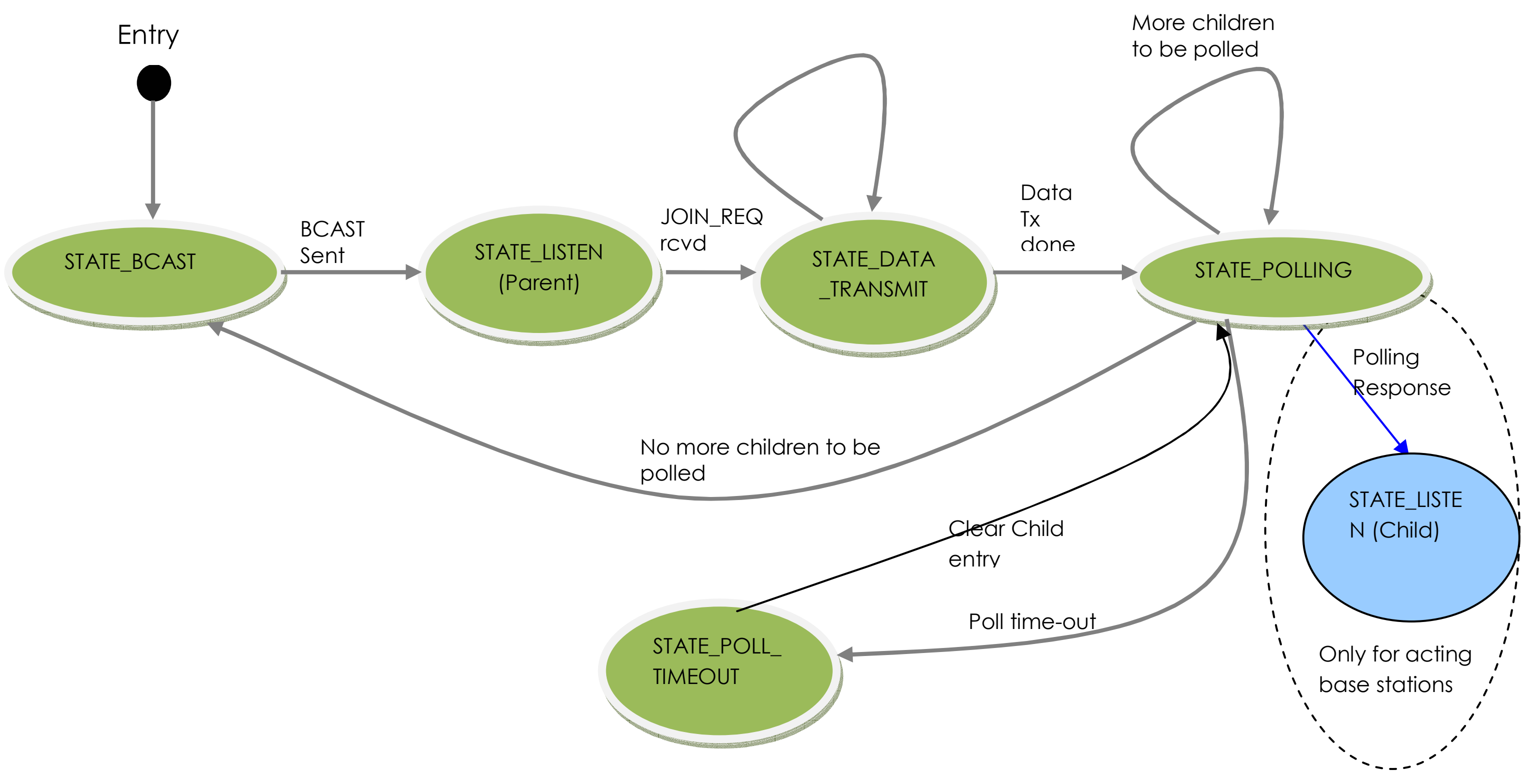


Figure-5: State transition diagram for Base-Station

The base station (root), after initialization enters the broadcast state (STATE_BCAST) as in figure-5, in which it transmits the Base-station Control Message (BCM). Once BCM is sent it moves to the listen state (STATE_LISTEN) where it listens for any incoming join requests (JREQs) from other nodes. The listen period can be varied as per our needs. If it receives the join request then the sending node is registered as child. After this it enters the data transmit state (STATE_TRANSMIT_DATA) where it transmits all the buffered data. Once the data transmission is done it enters the polling state (STATE_POLLING) where it polls each of its registered child nodes one after another. The polled child node can then carry out its tasks such as data transmission or polling of other nodes under it. If the registered child node does not respond to the polling message with a polling response message with a certain time, then the base station times out (STATE_POLL_TIMEOUT)and moves again to polling state to poll the remaining child nodes. Once all the child nodes have been polled the base station (root) enters the broadcast state again unless it is an acting base station (ABS). ABS move to STATE_LISTEN where it enters the child station role as explained below.

6.2. State Transition Diagram for Child/Acting Base-Station

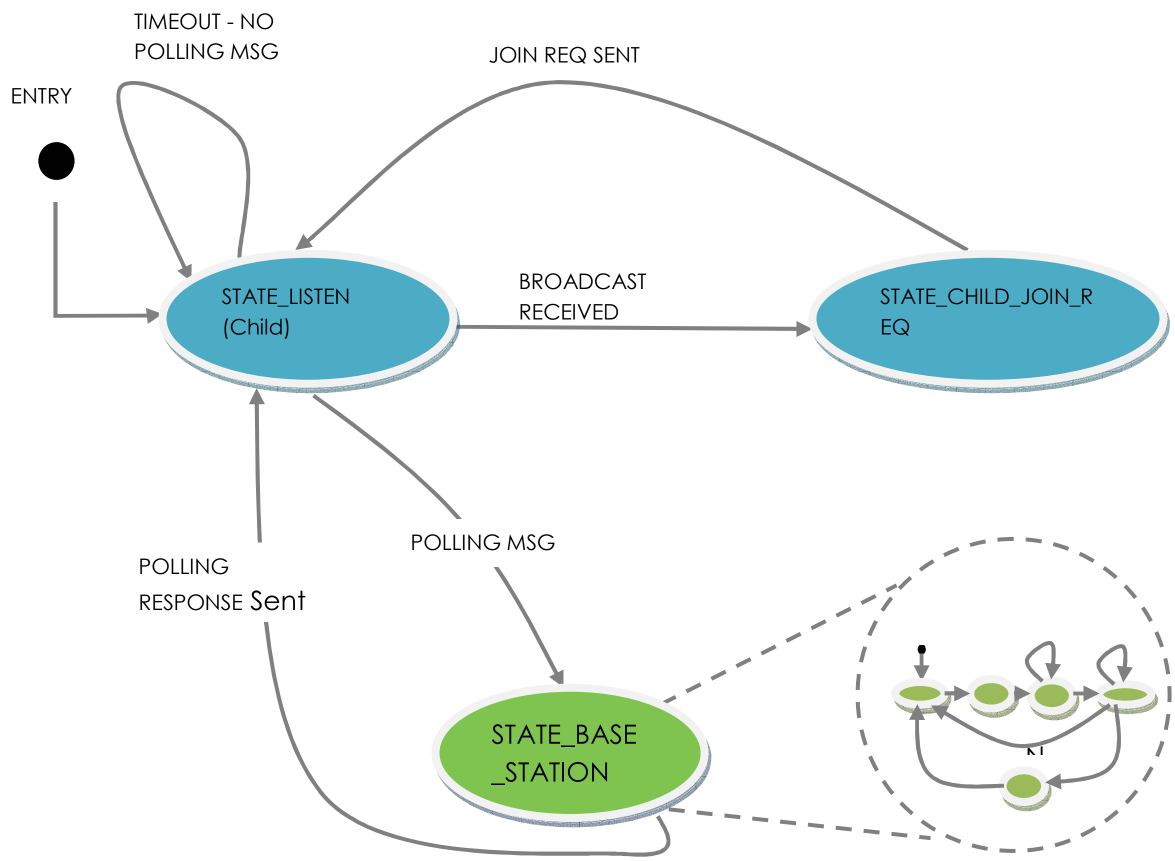


Figure-6: State transition diagram for the acting Base-Station

The states of other stations except the root follow the transition diagram as shown in figure-6. These nodes start in the STATE_LISTEN-(child) where they listen for any broadcast message from a BS. Once the broadcast message is received it enters the join request state (STATE_CHILD_JOIN_REQ). Here it sends the join request to its parent node. Then it moves back to the listen state and listens for the 'Polling Message'. If polling message arrives then it registers the sender BS as it's sole base-station. If BS registration doesn't occur, then it responds to other broadcast messages from nearby BS. If it receives the polling message from its BS, it enters the STATE_BCAST sub-state inside STATE_BASE_STATION and follows the same transition diagram for root as shown in figure-5. Once the polling is done it comes back to STATE_LISTEN-(child) after sending 'Polling Response Message' and continues to listen for another polling message.

7. TinyOS Components

Polling MAC protocol is implemented using the CC1000RadioC component which provides interfaces and methods to send and receive messages over RF. The existing CC1000RadioC wires the control and data paths whose implementation is hidden from the upper layers. CC1000Control handles the frequency and power control. CC1000RadioIntM handles the CSMA, encoding of data, preamble detection and synchronization.

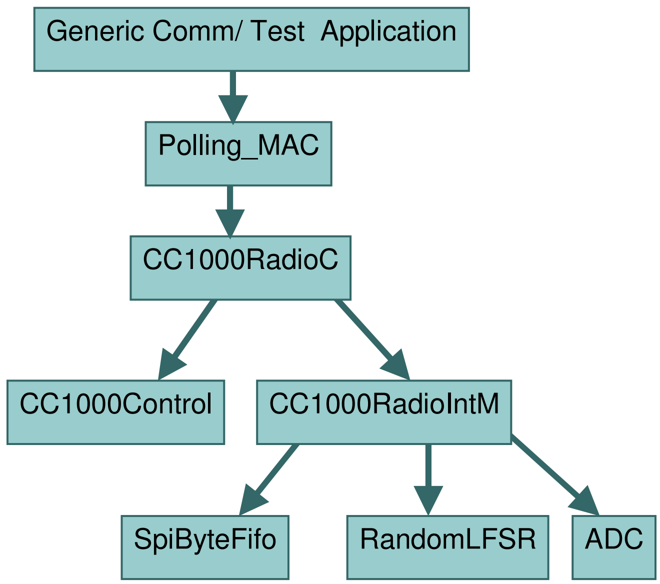


Figure-7: Block diagram of Components involved in Polling MAC protocol implementation

We are providing a PollingMAC component which implements the interface IPolling. IPolling interface contains the *commands* for enabling/disabling MAC layer, sending data, and *events* for receiving data and receiving status of data transmitted. IPolling also computes checksum, handles buffer for sending messages and manages various timers.

8. Features Implemented

Polling MAC protocol for a multi-hop network has the following features.

a) One of the motes is a Root base station (with TOS_LOCAL_ADDRESS=0) which receives data from all other motes and transmits to a computer using serial communication. The Root base station co-ordinates the polling of the entire network and doesn't transmit any user data.

b) The Root base station has the ability to sniff the nearby motes' messages and transmit to a computer using serial communication. This feature can be used for network monitoring and troubleshooting.

c) A base station (Root and ABS) implements a timeout mechanism to remove non-responsive child motes from the polling list and reuse the empty slots for new child motes. This timeout value can be customized based on the network requirements. Current implementation uses 6 seconds for timeout and 10 such consecutive timeouts are required to evict a child mote entry from the polling list.

d) A base station (Root and ABS) implements a timeout to make sure a child mote doesn't transmit data beyond a certain limit in each polling round. After the timeout period, base station starts polling the next mote in the list.

e) A child mote implements a timeout mechanism to recover from a base station failure. If no base station polling message is received, this timer expires and child invalidates the current base station entry and start listening for other base station broadcast messages. Current implementation uses a 30 second value for this timeout period.

f) A child mote waits for a random duration between 0 to MAX_LISTEN_DURATION (e.g 50ms) before sending join request messages to avoid collision at the base station.

g) A circular queue (FIFO) is implemented to buffer the sent messages. Current implementation uses a buffer size of 10 messages. This value can be customized to suit network requirements.

h) A CRC is computed and attached for transmitted data. Received data is sent to the upper layer only if CRC matches correctly.

i) Serial communication between mote and PC is implemented to collect debug logs. A circular queue of size 25 is used to buffer debug messages.

j) A test application is written over Polling MAC to measure the performance.

9. Testing Strategy

9.1. Topologies

Our implementation of the multi-hop protocol was successfully tested on the following topologies.

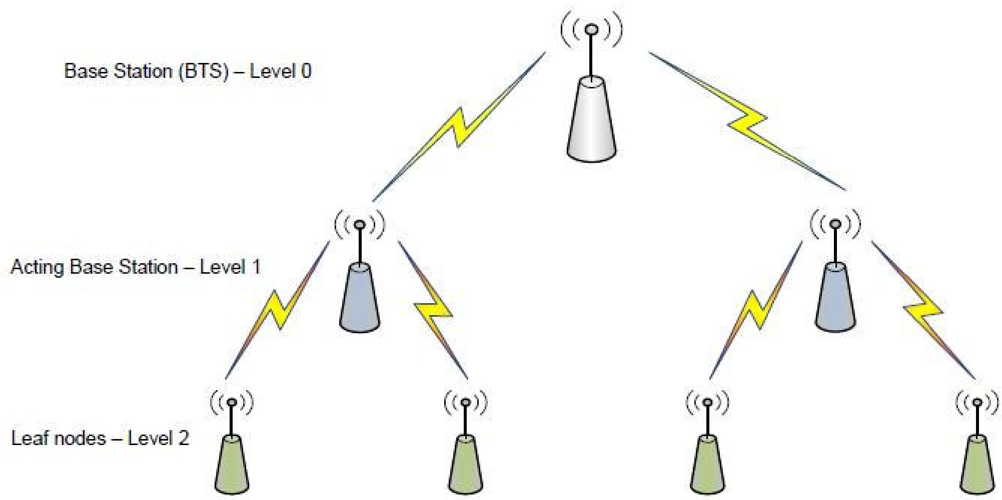


Figure-8: Hierarchical Network Topology – Binary tree structure

The above topology represents a hierarchical network in which the motes were arranged in a binary tree structure. In this topology, the Base-station (at level 0) has two child nodes, both acting as base stations and have two child nodes of its own. The level of multi-hop in this network is 2.

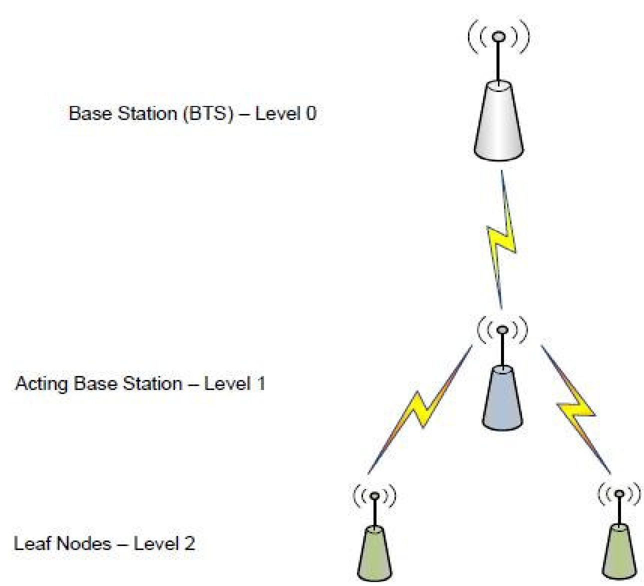


Figure-9: Multi-hop Network Topology – Two levels

The above topology represents a multi-hop network of two levels. In this topology, the Base-station (at level 0) has one child node, which in turn acts as base station (ABS) to its two child nodes. The level of multi-hop in this network is 2.

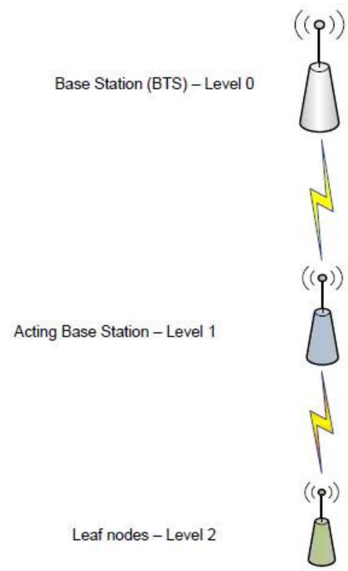


Figure-10: Multi-hop Network Topology – Two levels

The above topology represents a multi-hop network of two levels. In this topology, the Base-station (at level 0) has one child node, which in turn acts as base station (ABS) its child node. The level of multi-hop in this network is 2.

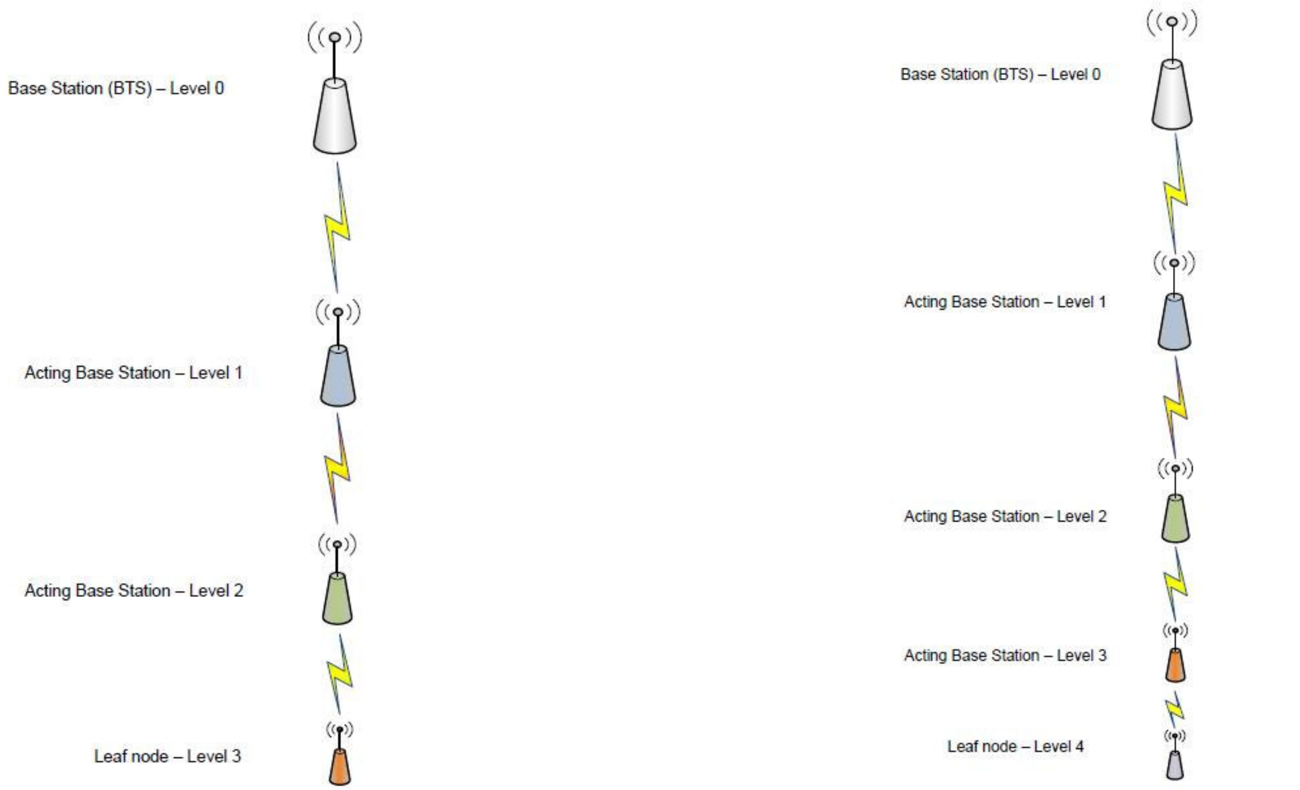


Figure-11: Multi-hop Network Topology – Three and Four levels

The above topology represents a multi-hop network similar to figure-10 but with three levels on the left and four levels on the right.

9.2. Test application

In order to test the Polling MAC protocol, a test application was developed which functions as a traffic generator. The traffic generator generates traffic at a constant bit-rate (CBR). The rate produced by the traffic generator can be modified by varying the parameter "TIME_PER_PACKET" in the code.

9.3. Test Setup

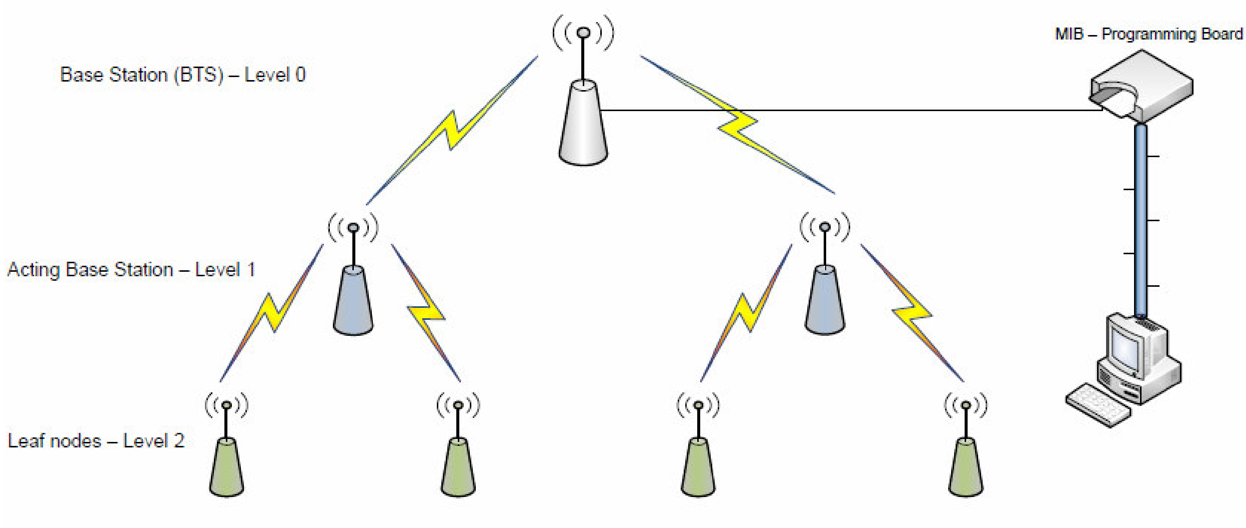


Figure-12: Test Setup

The test setup is as shown in figure-12. All the nodes send the packets to their base-station except Root BS. Each ABS node would buffer the packets sent by its child nodes and sends them to its parent BS and this continues till packets reach the Root BS. The Root BS transmits these messages and other statistics to a computer over a serial port.

In order to debug the system, a new function was written which would make use of the UART mechanism to transmit the debug messages to the remote PC. All nodes can transmit the debug messages. A mote needs to be placed on the MIB600 programming board connected to a PC to view the debug messages using telnet at port 10002. This would enable to debug and monitor the system.

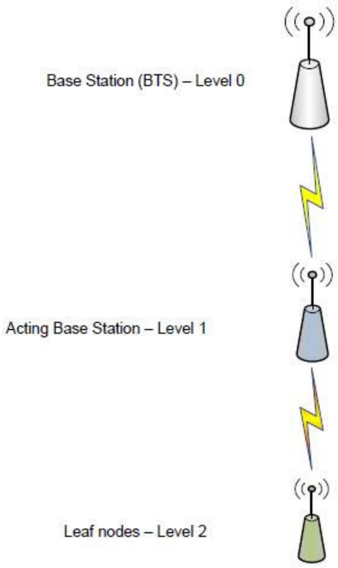
The throughput analysis of the protocol was carried out as follows. Each mote was flashed with the test binary to send data at a constant bit-rate to its corresponding base station. This in turn would make all packets reach the Root BS connected to a PC to gather results. All the motes had a buffer to hold at most 10 packets. At the root base station, each packet received was numbered and time-stamped. The analysis was carried out varying the load and noting down the average throughput achieved for various topologies.

The result of the throughput analysis is explained in the following section.

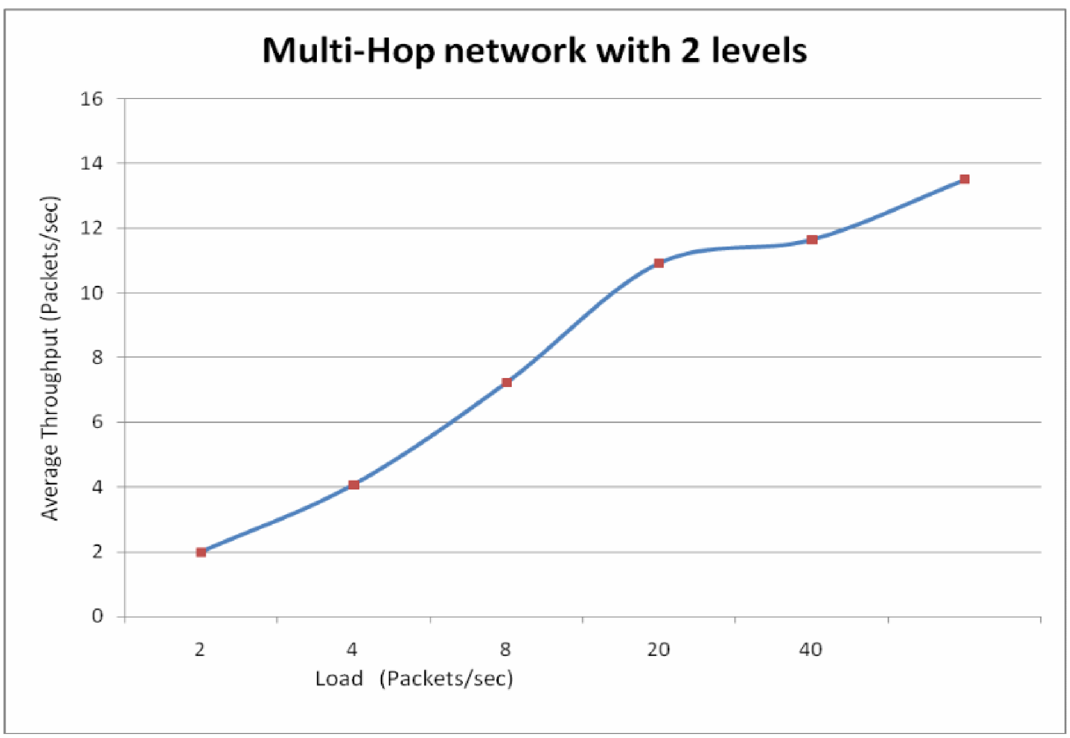
10.Results

10.1. Multi-hop Network with two levels

The network topology, results and the plot of average throughput is shown below.



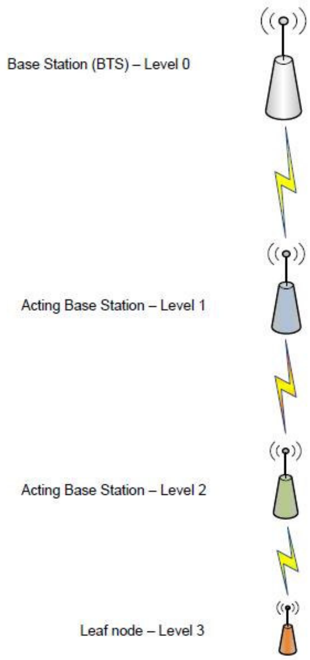
Load (packets/sec)	Average Throughput (packets/sec)
2	1.99371581
4	4.074403439
8	7.220558584
20	10.9229677
40	11.64711839
50	13.49702419



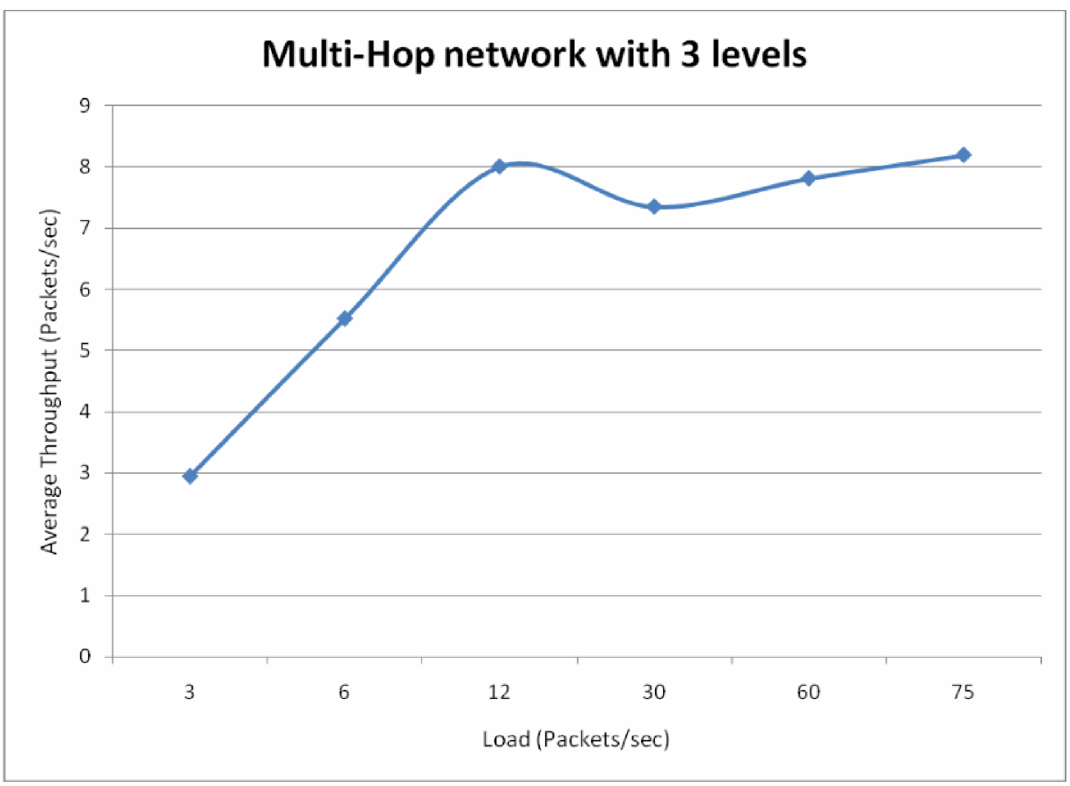
The mote at level-1 could send more than 10 packets per second to Root BS when each mote was sending data at more than 10 packets per second.

10.2. Multi-hop Network with three levels

The network topology, results and the plot of average throughput is shown below.

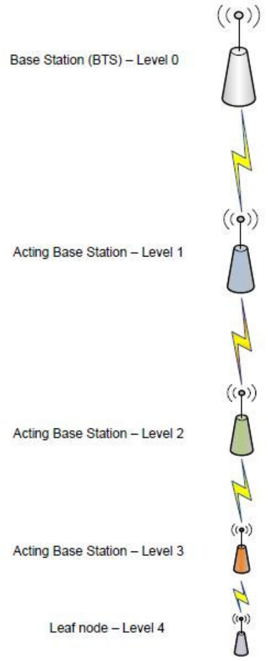


Load (packets/sec)	Average Throughput (packets/sec)
3	2.953413194
6	5.525608162
12	8.003881622
30	7.350701387
60	7.812297138
75	8.193093607

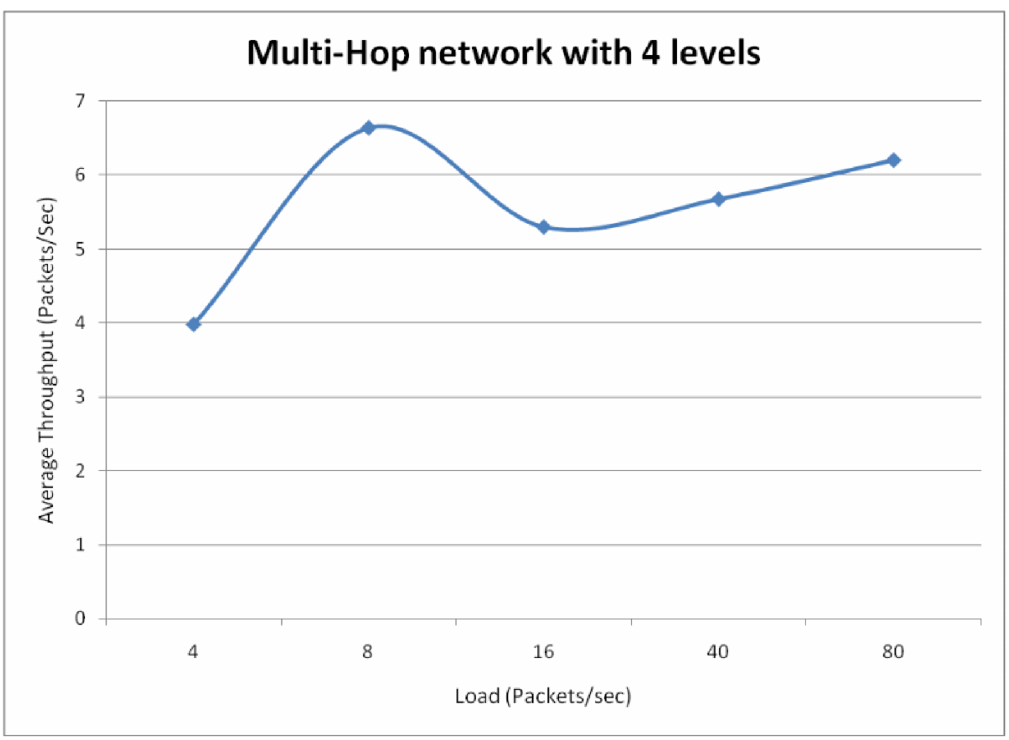


10.3. Multi-hop Network with Four Levels

The network topology, results and the plot of average throughput is shown below.

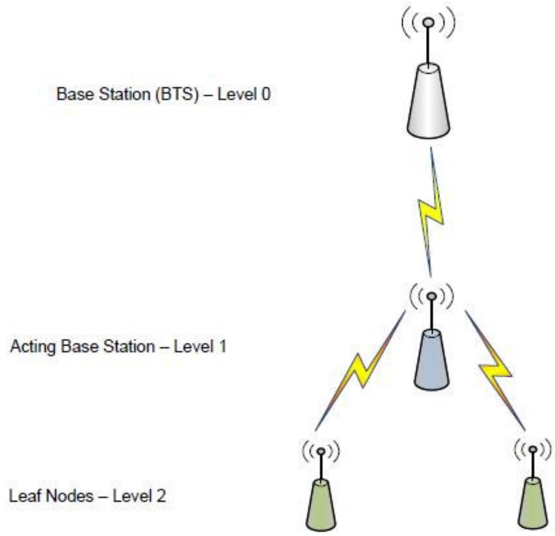


Load (packets/sec)	Average Throughput (packets/sec)
4	3.98063475
8	6.640856954
16	5.298352052
40	5.675248711
80	6.203739638

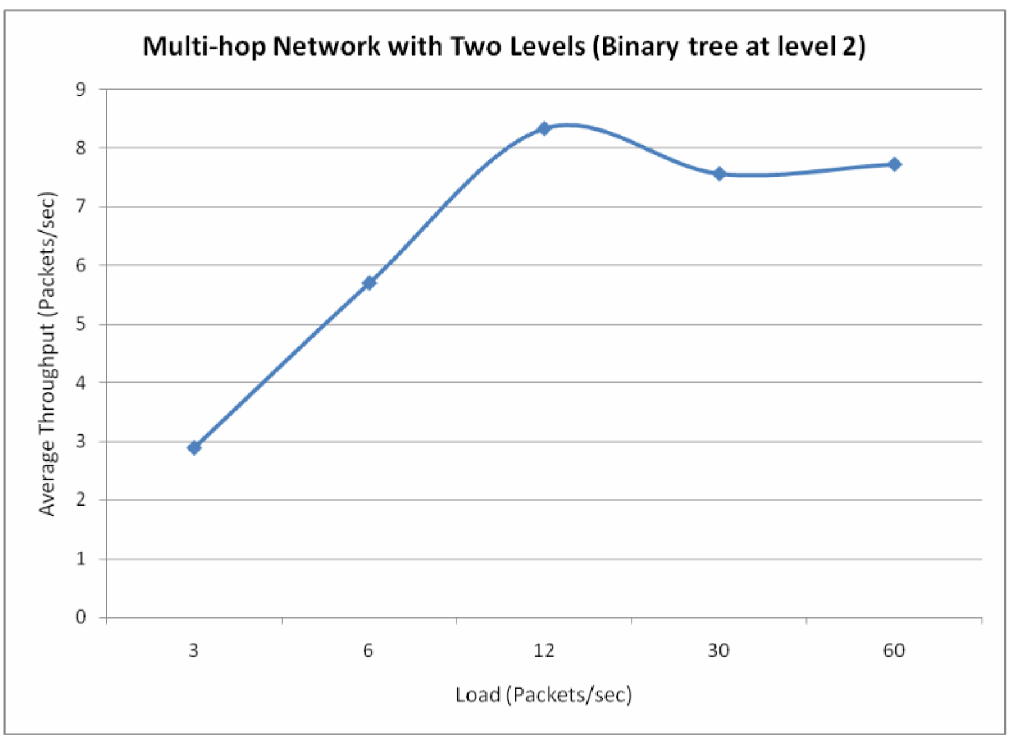


10.4. Multi-hop Network with Two Levels (Binary tree at level 2)

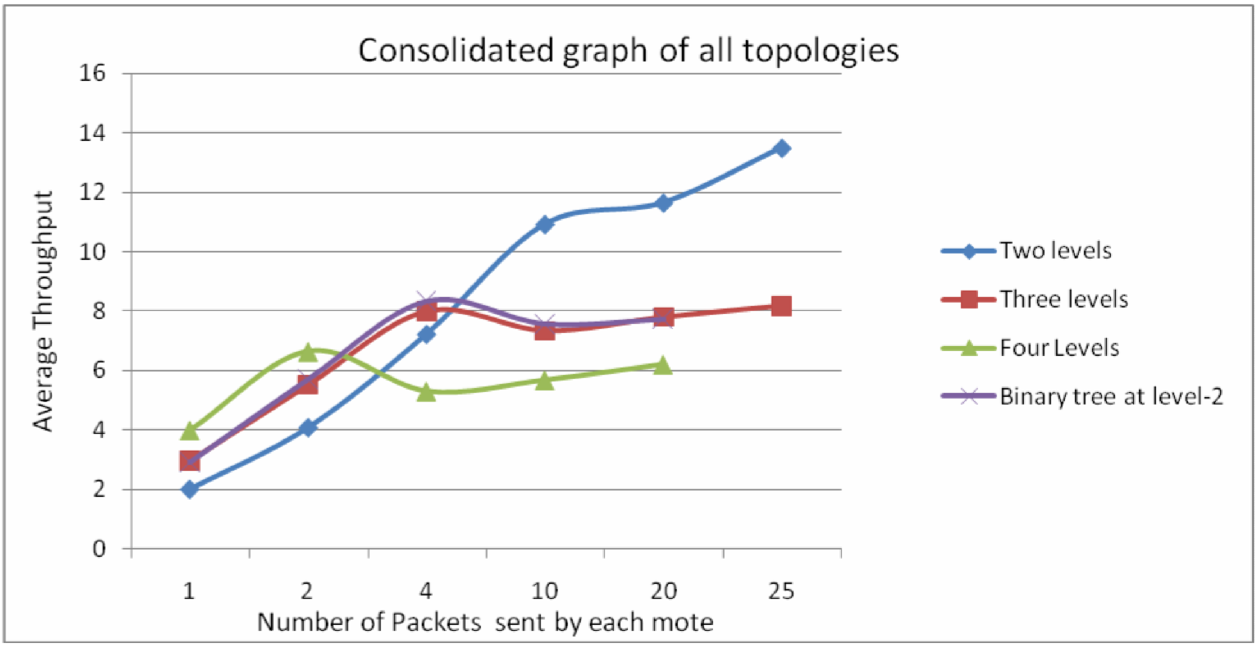
The network topology, results and the plot of average throughput is shown below.



Load (packets/sec)	Average Throughput (packets/sec)
3	2.895213215
6	5.703658358
12	8.335837354
30	7.565505313
60	7.726900964



10.5. Consolidated graph



10.6. Observations

- For a given topology, a linear increase in the average throughput is noticed initially with increasing load and the throughput remains almost constant thereafter and doesn't increase with increase in the load. A dip in curve is found when no. of packets received exceeds the buffer size as some packets get dropped.
- For a **light load**, throughput increased with the increase in no. of motes (hops). Most Packets were buffered at each level in light load scenario. The frequency of available transmission slot per mote increased with light load (polling overhead is less).
- For a **heavy load**, throughput decreased with the increase in no. of motes (hops). This is because of limited buffer space. Packets were dropped at each level when buffer was full. Also frequency of available transmission slot per mote decreased with heavy load (polling overhead is more). If the number of 'Acting base station's 'are less then, less amount of time is spent in polling and very few packets are dropped and hence the throughput increases.
- It has also been observed that we get a better throughput when the topology is as shown in figure-9 than as shown in figure-11. This is basically because in former we have only 1 acting base station but in the later we have two acting base stations.

11. Conclusion

Throughput of Polling MAC protocol in a multi-hop sensor network depends on the following factors.

- Topology (number of hops or levels and number of nodes at each level.)
- Buffer size of each node. (Amount of data a mote can transmit in each polling slot.)
- Listen Duration of BS and ABS.
- Transmission speed of mote.
- Radio Interference in the surroundings.

Our results show that fewer the levels better the throughput under heavy load. Multi-hop penalty increases with each hop. Higher the levels better the throughput under light load for a given buffer size at each node.

12. Task break-up based on Modules

Task	Description	Completion Date	Assigned	Status
1	Studying about various MAC protocols to analyze the complexities (about the chosen MAC protocol - pros, cons, feasibility, etc.)	02nd March	All	Completed
2	TinyOS build setup.	05th March	All	Completed
3	Inspect TinyOS hardware interfaces to implement MAC.	07th March	Jeevan	Completed
4	Study about integrating code with TinyOS MAC layer.	07th March	Sandeep	Completed
5	Study about the chosen MAC protocol in detail. (Implementation details)	07th March	Indira/ Varun	Completed

6	Design the components and interfaces required for MAC implementation.	12th March	Jeevan/ Sandeep	Completed
7	Make a high level design of the MAC protocol. (Messages exchanged, format, etc.)	12th March	Indira/ Varun	Completed
8	Base Station functionality	27th March	Sandeep	Completed
9	Transmitter and Receiver functionality	27th March	Jeevan	Completed
10	Traffic generator	27th March	Indira	Completed
11	CRC Computation and Back-off functionality	27th March	Varun	Completed
12	Basic testing and bug fixing.	30th March	All	Completed
13	Working model of MAC protocol.	5th April	Sandeep/Jeevan	Completed
14	Make testcases with Varun and Start report preparation.	10th April	Indira	Completed
15	Come up with a set of test cases for MAC protocol.	10th April	Varun	Completed
16	Complete Testing and Bug fixing	15th April	All	Completed
17	Completing Project report	17th April	All	Completed

References

[1] TinyOS: http://www.tinyos.net/tinyos-1.x/doc/tutorial/

[2] nesC: http://nescc.sourceforge.net/

[3] MAC Protocols for Wireless Sensor Networks: a Survey, Ilker Demirkol et al.

[4] www.tinyos.net/papers/nest-2003-10-20-radio.ppt

[5] Mobile Communications 2nd Edition, Jochen Schiller, Pearson Education Publications

[6] http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf