MESH PROTOCOL DESIGN AND IMPLEMENTATION FOR BLUETOOTH 4.1 DEVICES

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

In

Information Security

Submitted By

Nikita Seth

(801433019)

Under the supervision of:

Mr. Arvind Kumar Verma

Staff Engineer, ST- Microelectronics

Dr. Neeraj Kumar

Associate Professor, Thapar University

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

May 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*MESH PROTOCOL DESIGN AND IMPLEMENTATION FOR BLUETOOTH 4.1 DEVICES*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Arvind Kumar Verma* and *Dr. Neeraj Kumar* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Nikita

(Nikita Seth)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Arvind kumar verma

(Mr. Arvind Kumar Verma)
Staff Engineer, CCDS, ST-Microelectronics

Neeraj

(Dr. Neeraj Kumar)
Associate Professor, Thapar University

Countersigned by

Deepak Garg

(Dr. Deepak Garg)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. S. Bhatia)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEDGEMENT

It gives me a great pleasure to take this opportunity to thank **ST Microelectronics Pvt. Ltd**. and **Mr. Rakesh Malik** for giving me such a great opportunity to do project in their esteemed organization. I deem it my privilege to have carried out this dissertation work under this well-known quality conscious organization.

I would like to express my sincere thanks to my immediate supervisor **Mr. Arvind Kumar Verma**, for guiding me through this project. He has been a source of inspiration and has constantly inspired me to give my best and exert my capabilities to the fullest. Without his arduous task of reviewing my work at every step, the project could not have been completed in the present form. I would also like to thank **Mr. Pratap Narayan Singh** for inspiring me and providing ideas to accomplish the project tasks in a better way. I would also like to express my sincere thanks to **Dr. Neeraj Kumar** (**Thapar University**) for guiding me through this dissertation and enlightening me about the research approach to be followed.

My sincere thanks to all the senior professors of **Computer Science and Engineering Department**, **Thapar University, Patiala** especially the **HOD**, **Dr. Deepak Garg** and **Dr**. **Jhilik Bhattacharya**, for providing me an opportunity to take up this training and for their constant support and encouragement. At last I would also like to express special thanks for all my friends and family for their constant support and care.

NIKITA SETH

ABSTRACT

Bluetooth low energy (BLE) is a promising solution for implementation of low power and cost effective communication mechanisms especially in home automation, wearable, healthcare and sensor technology. However, limited range of BLE devices makes home automation using BLE less efficient. In order to utilize BLE capabilities in an efficient manner there is a need of mesh protocol which can support multi-hop communication. In this dissertation, the mesh protocol for Bluetooth 4.1 device s has been proposed (ST's BlueNRG-MS) devices. This implementation is a cost-effective solution for applications in home automation. The proposed algorithm works in two phases. In phase 1, Network is initialized forming a mesh. After network initialization, each node will have information about all other nodes in the network (up to 64 devices). This information will be used to dynamically determine the shortest path between two communicating devices. Also, simulation results shows that the proposed algorithm outperforms the existing solutions in terms of firmware size and reliability.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

SNO.	Abbreviation	
1	BLE	Bluetooth Low Energy
2	SIG	Special Interest Group
3	FHSS	Frequency hopping spread spectrum
4	PHY	Physical layer
5	LL	Link Layer
6	L2CAP	Logical Link Control and Adoption Layer
7	HCI	Host Controller Interface
8	ATT	Attribute Protocol
9	SM	Security Manager
10	GAP	Generic Access Protocol
11	GATT	Generic Attribute Protocol
12	T_x	Transmission Power
13	T_s	Scan Window
14	T_a	Advertising interval.

Chapter 1

INTRODUCTION

Bluetooth Low Energy (BLE) is an ideal wireless standard for IOT Applications. BLE, Bluetooth smart, Bluetooth smart ready are the buzz words in the market [1] [2]. BLE devices have already been adopted highly in health-care, fitness, wearable, sensors and mobile industries [3]. Implementation of BLE for home automation in cost and energy efficient manner is the next target of various research groups [4]. For home automation, BLE is much more low power and cost efficient solution as compared to other short range wireless communication protocols available in the market like Zigbee, Wi-Fi etc. [5]. Also, BLE is supported in mobile phones which is an add on as compared to Zigbee.

The major challenge, while utilizing BLE capabilities in home automation, is to allow communication between devices which are not in direct range of each other or not in range of controlling device. It requires the control packet to be sent from one node to other till it reaches destination forming a mesh network. An efficient routing mechanism is needed to support this multi-hop communication among devices. The available solutions in the market are based on broadcasting which doesn't require routing mechanism. This dissertation introduces the implementation of a mesh routing protocol for BlueNRG MS devices which are compliant with Bluetooth specification v4.1 [6]. These devices can be controlled through Bluetooth Smart Ready Phones or devices. The proposed algorithm works in two phases, in phase 1 network is initialized, and every node has information about existence of every other node. This data is used to determine the shortest path between source and device to control. The rest of this report is structured as follows: Detailed state of art has been discussed in Chapter 2. The sample network topology and problem formulation is discussed in Chapter 3. Chapter 4 explains various Packet formats for advertising and sending commands as well as routing algorithms. Proposed work has been compared with existing solutions and other proposed works in literature as well as evaluated our work in terms of number of packets required for communication depending upon the number of hops in chapter 5. Then work is concluded with future scope in chapter 6.

1.1 A Brief Introduction of Bluetooth Low Energy

BLE also known as Bluetooth smart originally designed by Nokia, was adopted by Bluetooth SIG as a part of Bluetooth 4.0 specification in 2010 with the design goals of having a radio standard with lowest power consumption at minimal cost and lowest bandwidth consumption [1] [7]. BLE operates in the 2.4 GHz ISM band, divided into 40 channels of 2 MHz spacing from 2402 MHz – 2480MHz in comparison to 79 channels of 1 MHz spacing in classical Bluetooth technology. Among the 40 RF channels, 3 channels are kept for advertisement i.e. 37, 38 and 39 and rest 0-36 channels are data channels. Advertisement channels are used to broadcast data. In BLE, the radio hopping is based on formula as per FHSS technique [7].

$$hop\_channel = (current\_channel + hop\_value) \bmod 37,$$

Where hop_value is based on connection parameters, different for every device.

The BLE protocol stack is completely different from classic Bluetooth. The three building blocks of protocol i.e. Application, Host and Controller are further sub-divided into different layers as demonstrated in figure 1.

1. **Controller**: This block contains the lower layers of the protocol i.e. PHY and LL and HCI, Controller part.
    a. PHY contains the actual circuit for analog communication. It also takes the control of the modulation, channel distribution, FHSS and demodulation and convert these analog signals to digital.
    b. LL is responsible for Real-time requirements of the protocol in addition to CRC check, Air protocol framing, AES Encryption. Also it handles the link state of the BLE radio. LL layer is responsible for defining roles:
        i. When not in connection:
            1. Scanner
            2. Advertiser
        ii. When in connection :
            1. Master

2. Slave [7]

Based on these roles, LL states can be scanning/listening, advertising, initiating/responding, connecting.

c. HCI allows communication between host and controller. They are the set of commands that allow interaction between them.

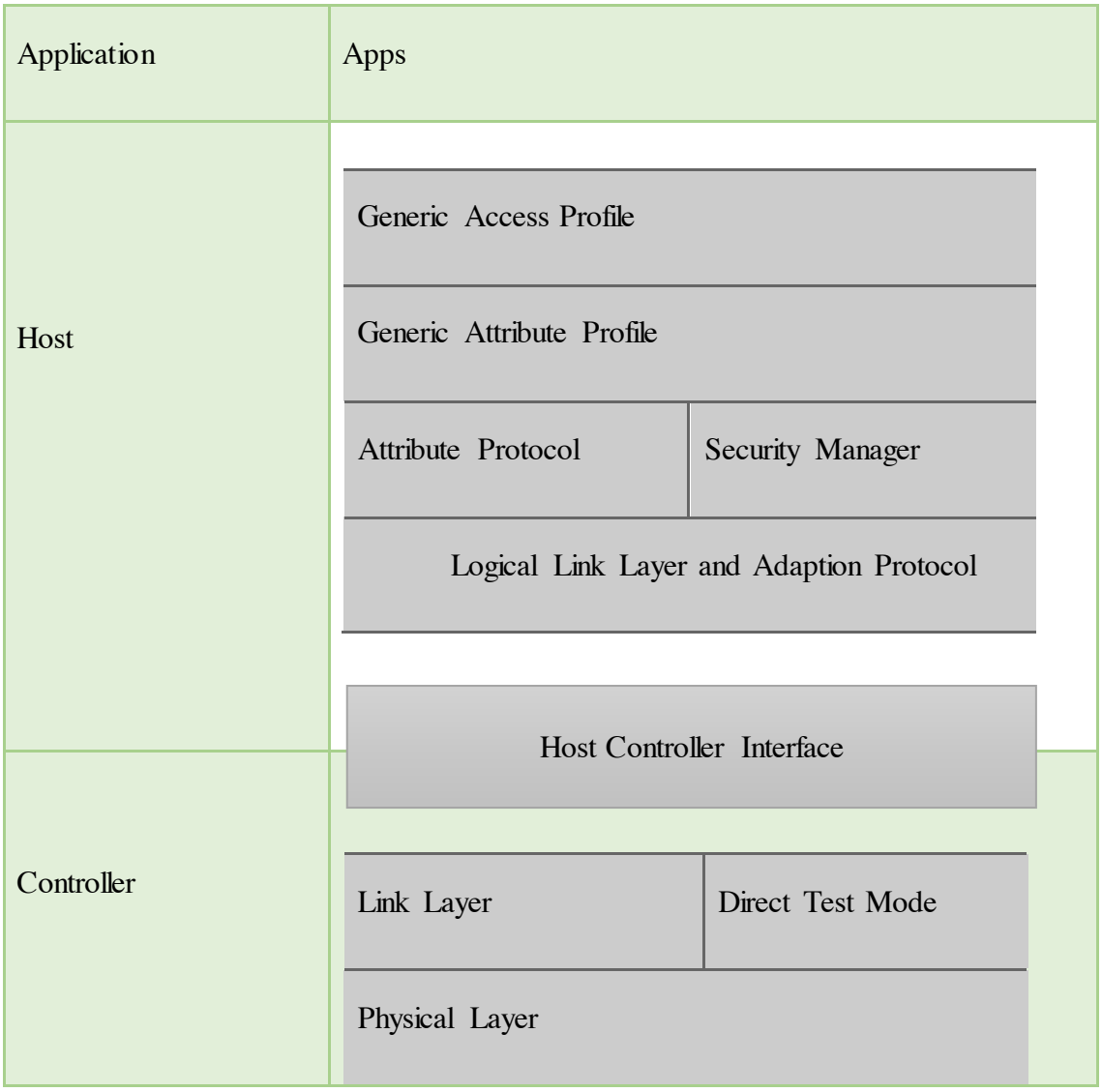

Figure 1: BLE Protocol Stack ([7])

2. **Host**: This block contains the upper layers of the protocol stack and is subdivided into 5 sub layers as follows:
    a. L2CAP is responsible for encapsulation of data from upper layers in a BLE packet format and vice versa. It also defragments and recombines large data chunks into 27-byte packet. It also takes care of upper layer protocol, ATT and SM [7].
    b. ATT manages client/ server requests based on attributes forming a stateless protocol for device communication. The devices can act as client/ server or both, and data at the server is organized in the form of attributes having a 16 bit handle called UUID. This UUID is used for communication. All read/ write procedures are handled by this layer [7].
    c. SM is the framework with series of security algorithms responsible for generating and distributing security keys among the devices for secure communication. It defines two roles for device
        i. Initiator (Master)
        ii. Responder (Slave) [7]
    d. GATT is built on top of ATT protocol, containing same client/ server model with encapsulated attributes as services which further contain characteristics. This layer provides generic profiles called GATT profiles [7].
    e. GAP is the outer most layer, defines low level parameters for control procedures like connection establishment, security management and device discovery [7].

3. **Application**: The highest layer responsible for user interface, data handling and other logics completely dependent on the user implementation [7].

GAP Roles and States

In contrast to LL roles, GAP defines 4 roles for device broadcaster (LL - Advertiser), Observer (LL - Scanner), Central (LL - master) and peripheral (LL - slave) [1] [7].

A Broadcaster/ Peripheral can have 3 states advertising, connection and standby as shown in Figure 2.

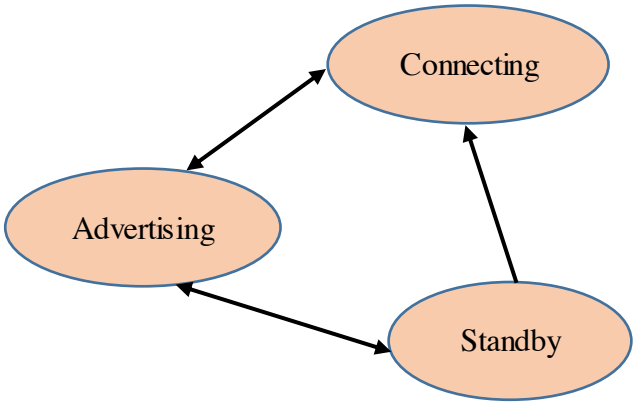

Figure 2: Broadcaster / Peripheral states

An Observer/ Central device can have 4 states i.e. Scanning, Stand By, Initiating and Connecting as in

Figure 3 [1].

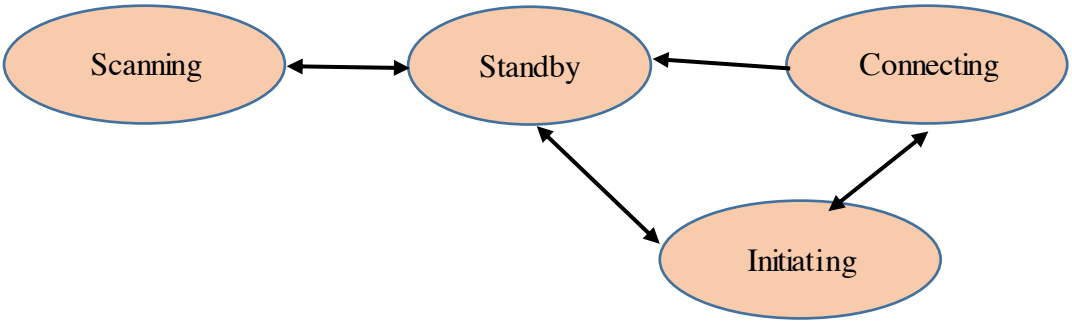

Figure 3 : Observer/Peripheral States

GATT Services, characteristics and descriptors

The attributes defined by ATT are organized in reusable and strict hierarchal structure in the form of services and characteristics. Similar attributes are grouped into services, which further contain zero or more characteristic. These characteristics can have further zero or more descriptors forming metadata as in Figure 4

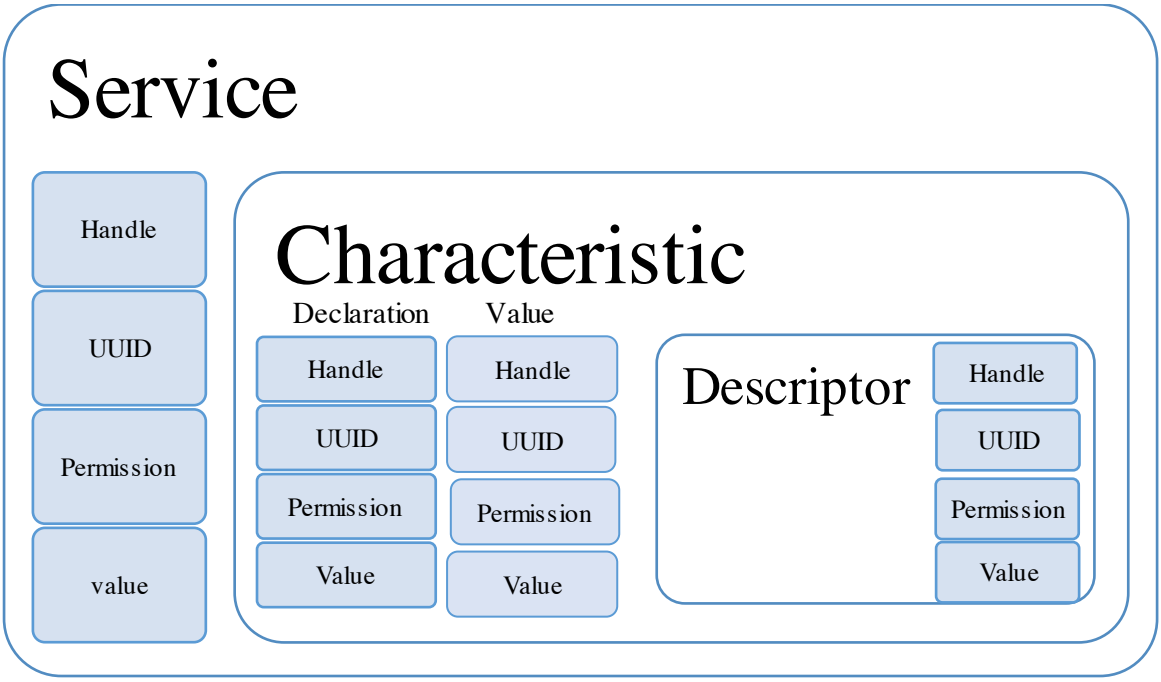

Figure 4: GATT Services, characteristics and descriptors [7]

Every Service has a uniquely identified handle using UUID having permission to read, read/write etc. Every characteristic has a handle for characteristic declaration and another handle which contains the value of the characteristic. For every characteristic there can be zero or more descriptors each which handle and UUID.

1.2 A Brief Overview of Android support for Bluetooth

Android Sdk provides Bluetooth APIs to handle Bluetooth based communication which allows scanning, transferring, connecting to Bluetooth devices [8]. All the APIs are available under the package android. Bluetooth. Bluetooth Adapter acts as the entry point to communicate with other Bluetooth peers [8]. It represents the local Bluetooth radio. These APIs provide all the functionalities required for communication. APIs like startLeScan and StopLeScan allow scanning of devices [8]. Similarly we have connectGatt API to connect to a particular device. Once the devices are connected, the characteristics can be read using Bluetooth service callbacks like onCharacteristicRead, OnCharacteristicWrite etc. [8].

In order to allow application to discover Bluetooth devices or play with Bluetooth settings etc., the application needs to have Bluetooth permissions. This can be done by adding below lines of code into the manifest file of the application.

<uses-permission android: name="android.permission.BLUETOOTH"/> [8]

<uses-permission android: name="android.permission.BLUETOOTH_ADMIN"/> [8]

1.3 Overview of Implementation setup

Figure 5: X-NUCLEO-IDB05A1 – BlueNRG-MS ([9])

BlueNRG-MS in Figure 2 is the board used for BLE implementation Features [9]:

- Bluetooth v4.1specification compliant master and slave single-mode BLE network processor
- Embedded all layers of BLE protocol stack
- Provides BLE profiles separately
- Operates at a supply voltage from 1.7 V to 3.6 V
- Max T_x Current is 8.2 mA (@0 dBm, 3.0 V)
- 1.7 μA of current consumption with active BLE stack
- Integrated linear regulator and DC-DC step-down converter
- Available output power up to +8 dBm (at antenna connector)
- Up to 96db RF Link
- Accurate RSSI to allow power control
- Proprietary application controller interface (ACI), SPI based, allows interfacing with an external host application microcontroller [9].

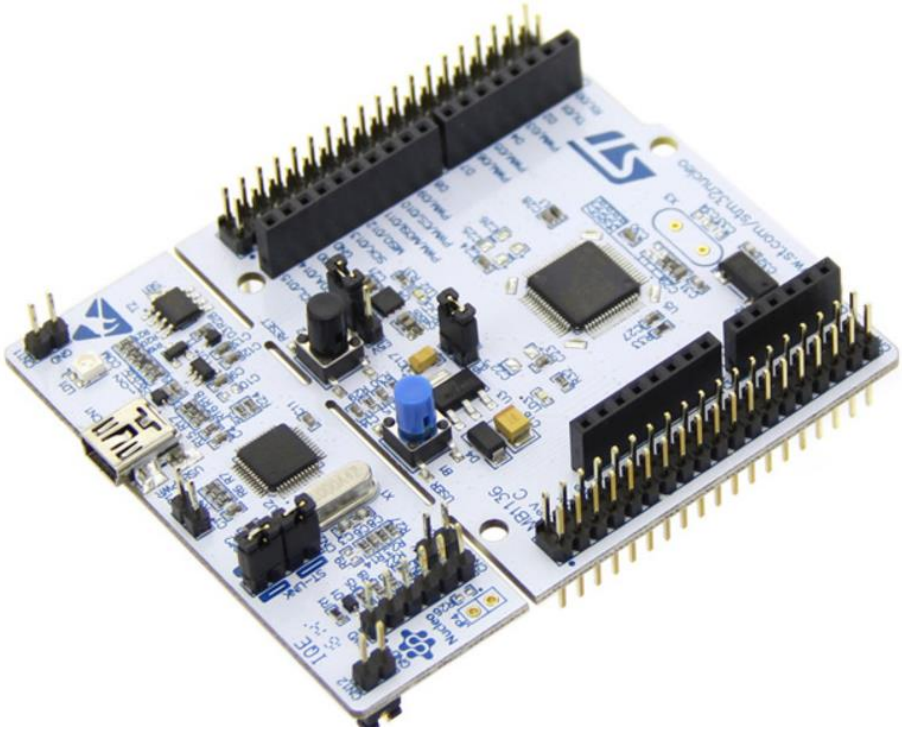

Figure 6: NUCLEO - L152RE ([10])

Figure 3 depicts the nucleo board used for debugging. Board in Figure 2 is and expansion to this board.

Chapter 2

LITERATURE REVIEW

2.1 State of Art on Comparison Analysis of available wireless protocols for communication

A lot of research has been presented in past providing the performance analysis of BLE among the other available short range wireless communication protocol like Wi-Fi, Zigbee etc. A brief summary of research works discussing how BLE is a best available solution among other protocols is presented below.

Mackenson et al. analyzed the performance of BLE sensor system in detail, comparing the allocated space of TI BLE Stack (CC2540F256) with Zigbee and classical Bluetooth, discussing that the T_x has no impact on the lifetime of the BLE sensor [11]. Mackenson et al. in [12] presented the feasibility of BLE based wireless sensors stating clearly that BLE consumes low power and good data throughput. Dementyev et al. analyzed the power consumption of BLE, Zigbee and ANT Sensor nodes considering the cyclic sleep [13]. They presented that actual power consumption depends upon the number of factors like average receiving power, transmission power, sleep current, awake current data rates and low duty cycle of BLE. They showed BLE consumes lowest power with Zigbee and ANT next in the list. Liu et al. in [14] demonstrated the energy consumption analysis if BLE advertising device and revealed a lot of facts that can be helpful while practically using BLE. For e.g.

For BLE scanner, $T_s > T_a$ and $T_s < 100\ ms$ will result in energy wastage

For BLE Advertiser, $T_a < 1000\ ms$ will consume more energy. Also, increasing T_a exponentially would lower the energy consumption [14].

In [15], Zhao et al. compared the BLE and Wi-Fi stating clearly how BLE is a better option than Wi-Fi for indoor localization because of various features of BLE like Channel hopping, a higher sampling rate that Wi-Fi and also lower T_x. Shahzad and Oelmann in [16] compared Zigbee, Bluetooth and Wi-Fi and stated the difference in the tabular form, clearly stating that BLE results in minimum energy consumption. Mikhaylov in [17] analyzed the performance if BLE at network level by creating their own simulator using MixiM framework. In [18], Lee et al. presented the anomaly in performance of device discovery in BLE stating that more frequent advertises lead to delay in neighbor discovery. Whereas, in [19] Jin-Shyan Lee et al. dis preliminary study on BLE, Zigbee and Wi-Fi and could not come to any conclusion which is better among the three. Also, Marco et al. in [20] compared the IEEE802.11ah standard with BLE and stated that BLE still needs to work for long range and Mesh topology.

A Comparison table of BLE, Zigbee and Wi-Fi, based on these works is shown below.

Table 1: Comparison of Short Range Wireless Protocol

Parameters	**Classical Bluetooth**	**Zigbee**	**Wi-Fi**	**BLE**
IEEE Specification	802.15.1	802.15.4	802.11 b/g/n	802.15.1
Frequency Spectrum	2.4 GHz	868/915 MHz; 2.4 GHz	2.4Ghz, 5Ghz	2.4 GHz
Topology	Star and P2P	Star, Mesh and cluster tree	Star and P2P	Star and P2P
Network Size	7	65536	32	Not Defined
Raw Bit Rate(Mbps)	1-3	0.02-0.25	11/54/600	1
Range (m)	10(class 2) , 100(class 1)	10-100	100 - 250	100
Number of channels	80	1/10/16	11-14 (3 orthogonal)	40
Security	Application layer/user defined	128 AES	SSID	128 AES
Relative Power consumption	Medium	Low	Low	High
Sample Battery life	Days	Months -Years	Months - Years	Hours

New features in BLE 4.1

BLE 4.1 specification has a lot of new features in comparison to previous version i.e. BLE 4.0 making it more superior for example:

1. Multi-mode support (devices can act as master and slave at the same time)
2. Low duty cycle
3. 32-bit UUID Support in LE
4. L2CAP Connection Oriented Channels and many more.
5. Eliminates overlapping signal interference of Bluetooth and 4G (LTE).
6. Allows manufacturer to specify the reconnection timeout intervals for their devices

Advances in Industries using BLE

With the advent of BLE, it has made up a strong presence in the marketplace. Healthcare, wearable, fitness & wellness, sensors all are using BLE in some way or the other [3] [2]. A few of the recent advances in these industries using BLE are depicted here in below table

Table 2: Recent Advances in industry using BLE

Sno.	Year	Industry	Paper Description
1	2014	VANETS	Frank et al. proposed the use of BLE for V2V communication. They suggested that using a smartphone and a BLE radio information can be communicated from one vehicle to other in case of immediate danger [21].
2	2015	VANETS	Jiun-Ren Lin proposed a blind zone alert system which monitors the blind zone of the vehicle and alerts the driver in a timely manner to prevent collisions. Their work proved a detection rate of 95%−99% and false alarm rate was < 15% [22].
3	2015	VANETS	Giin Lee et al. presented a smart watch based Driver Vigilance Monitoring System. In their proposed work, the PPG sensor is integrated in

			a sport wristband with a BLE module and transmit PPG signals to wristband/smart watch in real time. The estimated accuracy rate was 96% [23].
4	2014	Smart Lightening/ Smart Building	Seek Cho et al. presented a smart lightening system using a multi–sensor module which includes an ambient light sensor, temperature sensor, and a motion sensor along with a BLE module. [24]
5	2015	Smart Lightening/ Smart Building	Grover et al. presented a BLE based office automation system. They compared the available BLE modules by TI and ST and Used TI's CC250 for their implementation [25].
6	2015	Smart Lightening/ Smart Building	Choi et al. demonstrated an energy management system for smart office to reduce the energy consumption of PCs and Lights using BLE beacons and a mobile App [26].
7	2015	Healthcare	Mei-Ju Su et al. developed an ibrace Monitor system a Spine Care Service for old age patients of Osteoporosis. It integrates EMG and sensors for pressure analysis and position analysis and APP with cloud spine care service platform [27].
8	2015	Healthcare	Ye Ding et al. designed a Home Monitoring System for providing rehabilitation to patients at home or outside of hospital using wearable, BLE and IOT [28].

2.2 Study of Existing Bluetooth Low Energy Solutions in Home Automation

In order to utilize Bluetooth 4.1 capabilities in home automation, interconnection of nodes is a must. A few works using techniques like flooding, opportunistic routing and scatter-net formation have been discussed in past as given in below table.

Table 3: Existing solutions in Home Automation

Technique	Related Works
Flood Routing	Smart Office Energy Management System Using BLE Based Beacons and a Mobile App [26].
	CSRmesh- flooding protocol with close to zero setup time [29] [30].
Conventional Method (Scatter-net Formation)	Enhanced Bluetree -a Slave/Slave Mesh and Master/Slave mesh models using a 2 phase algorithm [31]
	On-demand Scatter-net formation - piconets are formed and as the nodes work in dual mode, thus forming scatter-net (mesh network) [32].
	FruityMesh- in this nodes perform advertising and scanning simultaneously and piconets and scatternets are formed. Connections are maintained throughout. [33] [34].
Opportunistic Routing	BLEmesh - based on opportunistic routing by broadcasting non-connectable advertisement data in batches. [4]

Chapter 3

PROBLEM STATEMENT

3.1 Gap analysis between existing work and desired work

Primary requirement for home automation is to access devices from any corner of the home. With Bluetooth being a short range communication protocol, it gets difficult to communicate with devices not in range. In order to utilize Bluetooth 4.1 capabilities in home automation, interconnection of nodes is a must. In case of flooding the broadcast packets are sent blindly which makes the communication very fast but there is no 100% guarantee of packet being received to destination, moreover, it may exhaust a lot of packets. A lot of applications in healthcare, intra-vehicular communication and office automation are based on beacon and broadcasting technique. But implementation of this technique for sending controls in home automation consumes a bit more energy and thus increasing the cost of devices. To overcome the shortcomings of these techniques and to have a cost-effective solution for home automation, we proposed an optimized approach for a mesh network formation among the devices. Mesh network is a basic multi-hop topology where all nodes communicate with each other. This requires devices to have information about each other's presence and thus send the control command from a device in range of controller to device not in range of controller. In our approach, network initialization is performed which is a one-time setup in case the devices are not reconfigured/ displaced and no new device is added into the network. Once the network is setup, devices can be controlled easily.

Chapter 4

PROPOSED WORK

4.1 Assumptions and Implementation Parameters

A network of 12 devices is considered as in Figure 7 but, the algorithms are designed for up to 64 devices in a network and are expendable for more. Dotted Lines in the diagram depict that the devices are in each other's proximity, for instance B, C and D are in proximity of A and C, G and E are in proximity of D and vice versa. All devices are configured with unique device ID within the network, a network name and network authentication key. Device ID ranges from 0 to 63. Network name length can be set to maximum 6 bytes. A is considered as the root node in Figure 7 signifying that A is in proximity of the controller. An android Phone with Bluetooth 4.x support acts as a controller in this implementation. The devices operate in dual mode i.e. can act as master and slave node simultaneously as per the Bluetooth 4.1 specification.

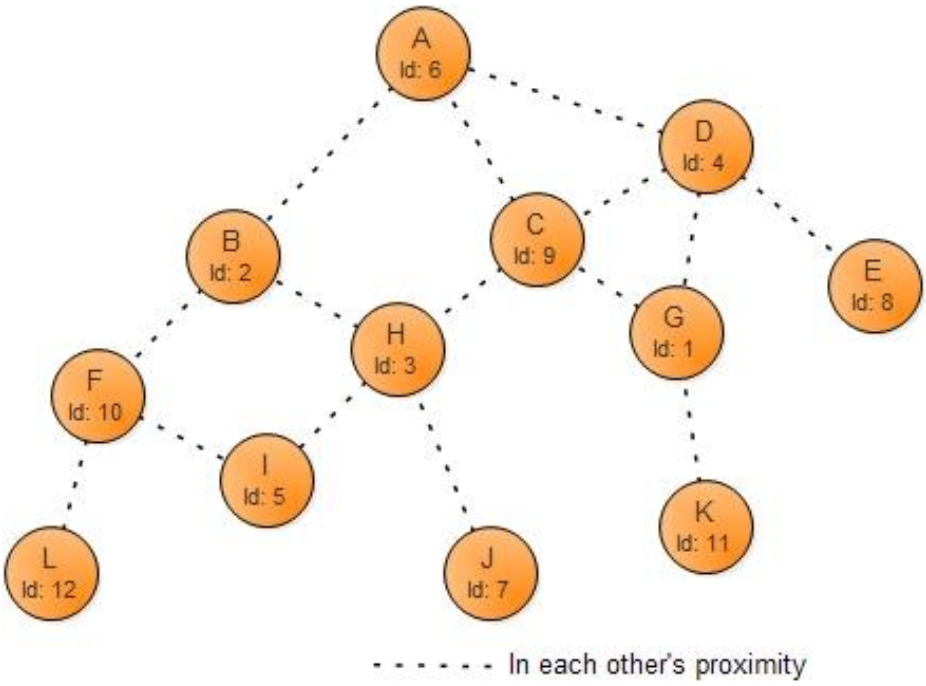

Figure 7: Sample network topology

Every device defines a custom GATT service which has three characteristics to perform different task.

1) 1st characteristic: It is defined as meshCommand characteristic with 20 byte data length. We use it to send different type of request to device mainly scanning/ initialization/ status update/ network data collection.

2) 2nd characteristic: It is defined as meshStatus characteristic with 20 byte data length. It is used to read status of current request.

3) 3rd characteristic: It is defined as meshData characteristic with 160 byte data length. We use this to provide connection matrix data to requesting device.

4.2 Methodology

In this proposed approach, the advertising parameters are set and scanning when idle in simultaneous mode. The advertising packet format is shown in Figure 8.

The Devices the filtered on the basis of Network name and ID. The advertisement packet of device A for example contains the information of all devices which are in proximity of A in Connection Matrix (8 Bytes).

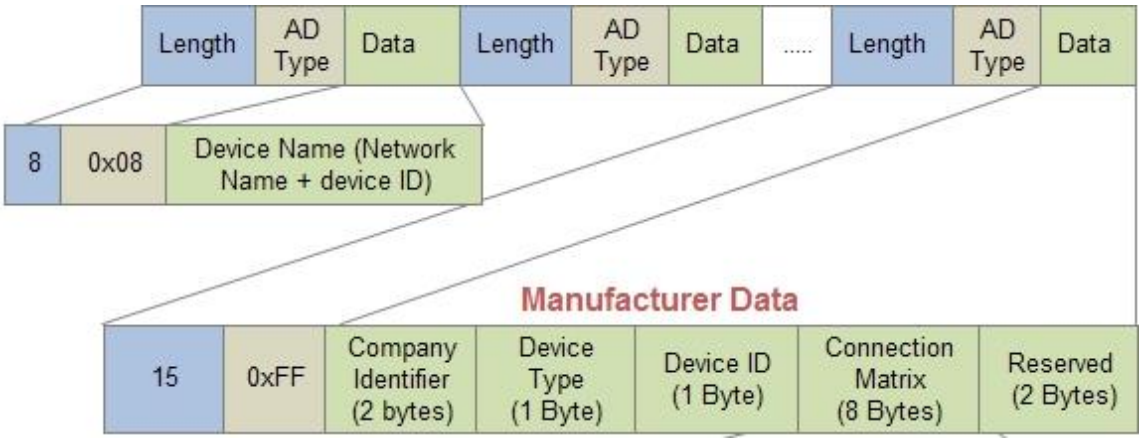

Figure 8: Advertisement Packet Format

Initially, the bitmap for A in Connection Matrix is all 0 bytes except one for A itself. When A is in scanning mode and advertising packet of B is received, A's connection matrix value gets updated with B's information. This depicts that B is in A's range. And same in case when C appears in A's range while A is in scanning mode. Devices maintain each other's information and form a mesh network at the time of initialization. This network initialization request is sent from the controlling device, in our case android phone, the algorithms for the same are discussed in chapter 4.3. Once the network is initialized the controller should be able to control the devices to switch on / off as explained in chapter 4.4. The command packet format used for sending network initialization request as well as control request is shown in Figure 9. The command packet for three different commands is given Init Command – Scan Request (for network initialization start request), Init Command – Data Update Request (for distributing network information among all devices in the network) and Control Command (for controlling the device to switch ON/OFF)

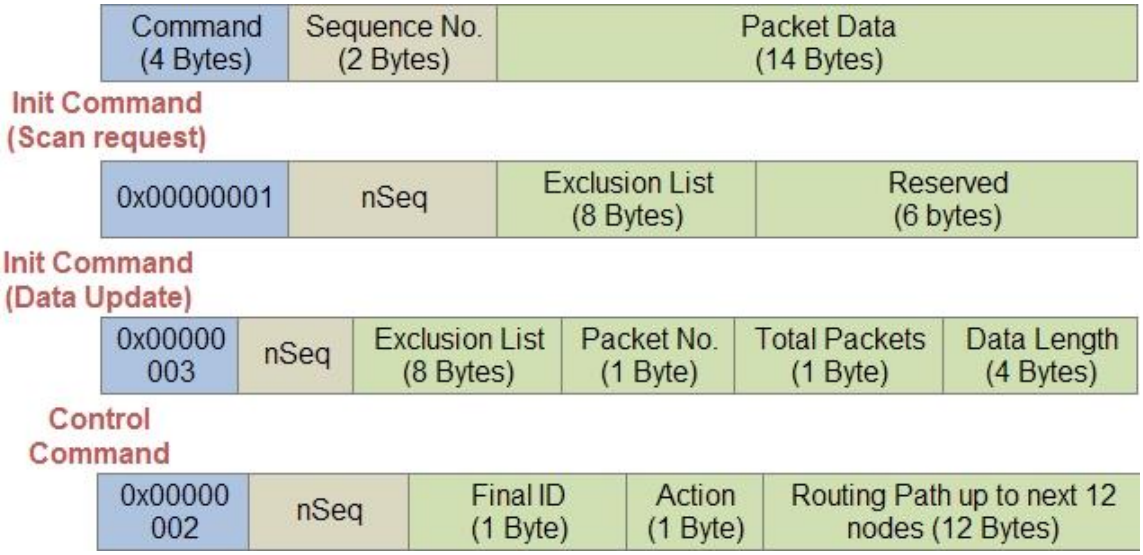

Figure 9: Command packet

4.3 Phase 1: Network Initialization

Network Initialization involves collecting data from each device and distribute it to every device in the network. In this step, controller connects to one of the visible devices and send initialization command with 8 byte exclusion list as data payload

Data collection Algorithm

This algorithm presents how initRequest is sent from controller to other devices. The devices poll for some fixed time interval _ and wait for init done status. This is one part of network initialization where data is collected at the root node and the requested node. The purpose of maintaining exclusion list is to avoid some of the duplicate requests. As every node has information about its children nodes we can avoid duplicate request till next two nodes in the chain. This algorithm takes two parameters as input one if InitRequest is received and other NextDeviceId i.e. the id of device to which packet is sent next.

For example, Node A receives initialization request from a controller with an empty exclusion list. A adds itself and its children B, C and D to this list. A has 3 paths to start searching and different exclusion lists is generated for each path through B, C and D. Using the above method first node B has exclusion list as

Command from A-> B = [ABCDGE]

(In actual command packet bits representing the node ID are set).

Note that common child node H of B and C is not included in exclusion list of B. So B can communicate with all its children nodes. We remove H in exclusion list of next node C since it was already covered in path through B.

Command from A -> C = [ABCDEHF]

Command from A ->D = [ABCDHFG]

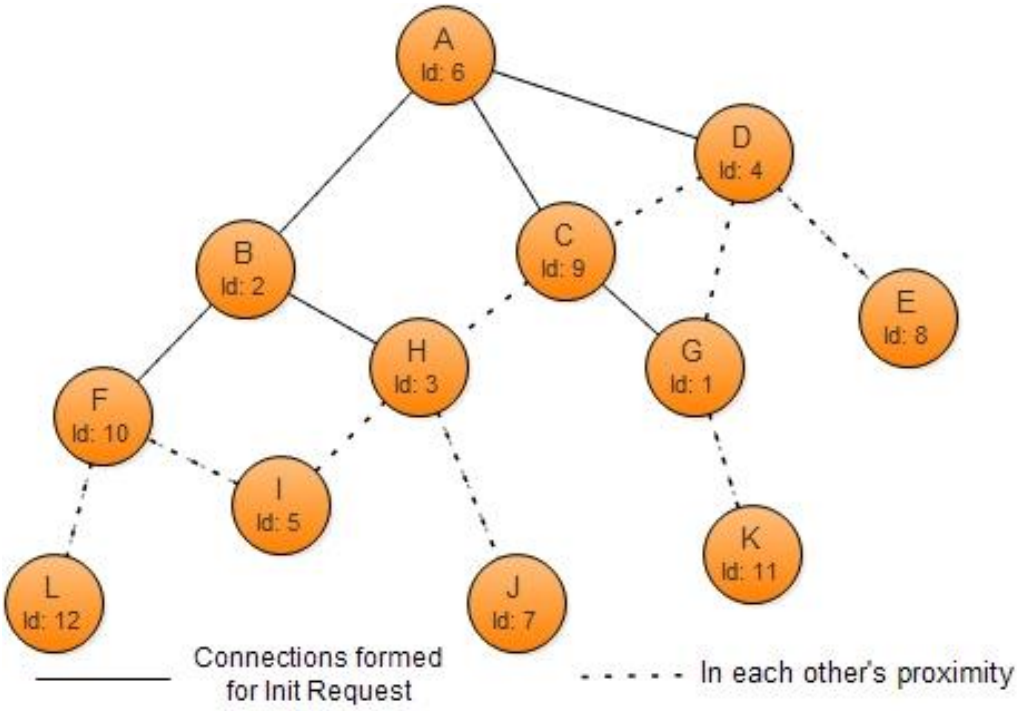

Figure 10: Network Path followed for sending Init requests

Figure 10 depicts the path followed for sending initialization request using the above mentioned algorithms. Thus at first level we have avoided redundant init request from [C -> H], [C -> D], [D -> C], [D -> G]. Similar decision is made at subsequent nodes and exclusion list is updated with all the nodes covered or to be covered in alternate paths until the leaf node is reached. When init request reaches node L, L has no subsequent nodes to forward request packet, then L updates its status characteristic as init done. Its requester node (F) will read the status (either by polling or by notification) and then update its meshData characteristic by reading L's meshData characteristic. In a similar way connection bitmap data of each node is propagated back to first Initiator node, A in this case.

Data Distribution Algorithm

Once the data is received at controller end, it needs to be distributed to each device in network. This data distribution is done using same algorithm as in case of data collection with the difference that data is updated at meshData characteristic of the slave node along with init step 2 command (data update request) packet.

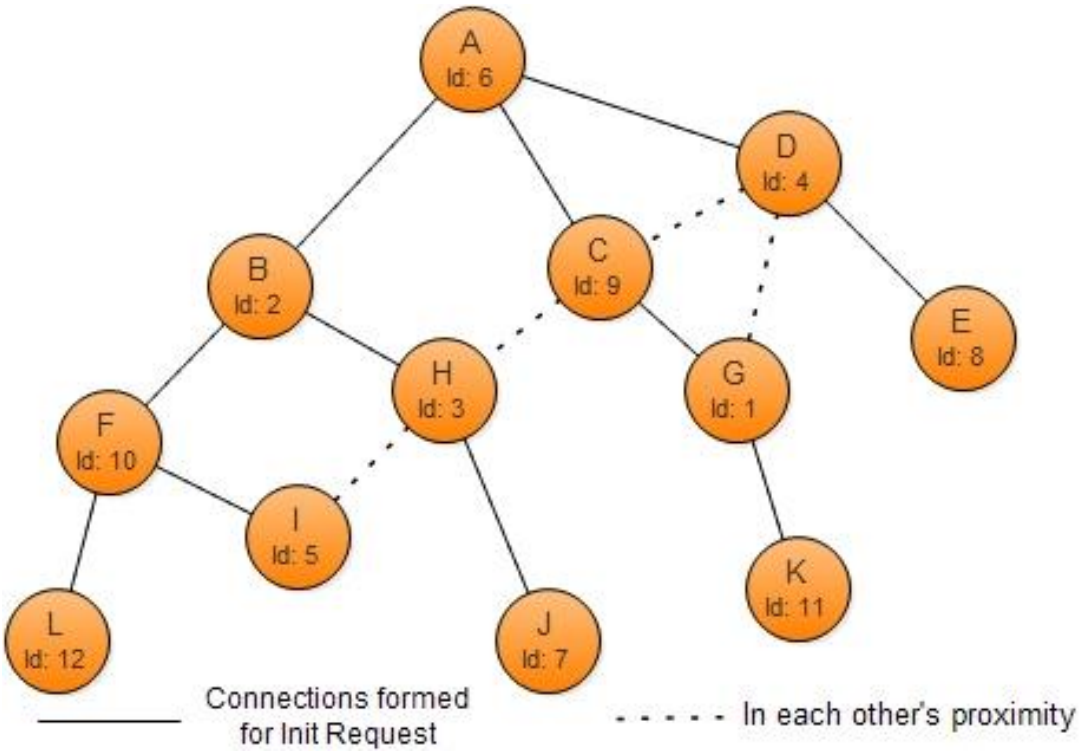

Figure 11: Network path followed for data distribute request

Figure 11 presents the connections required for data distribute requests. As leaf nodes should also get the data, so connections are to be made till leaf node.

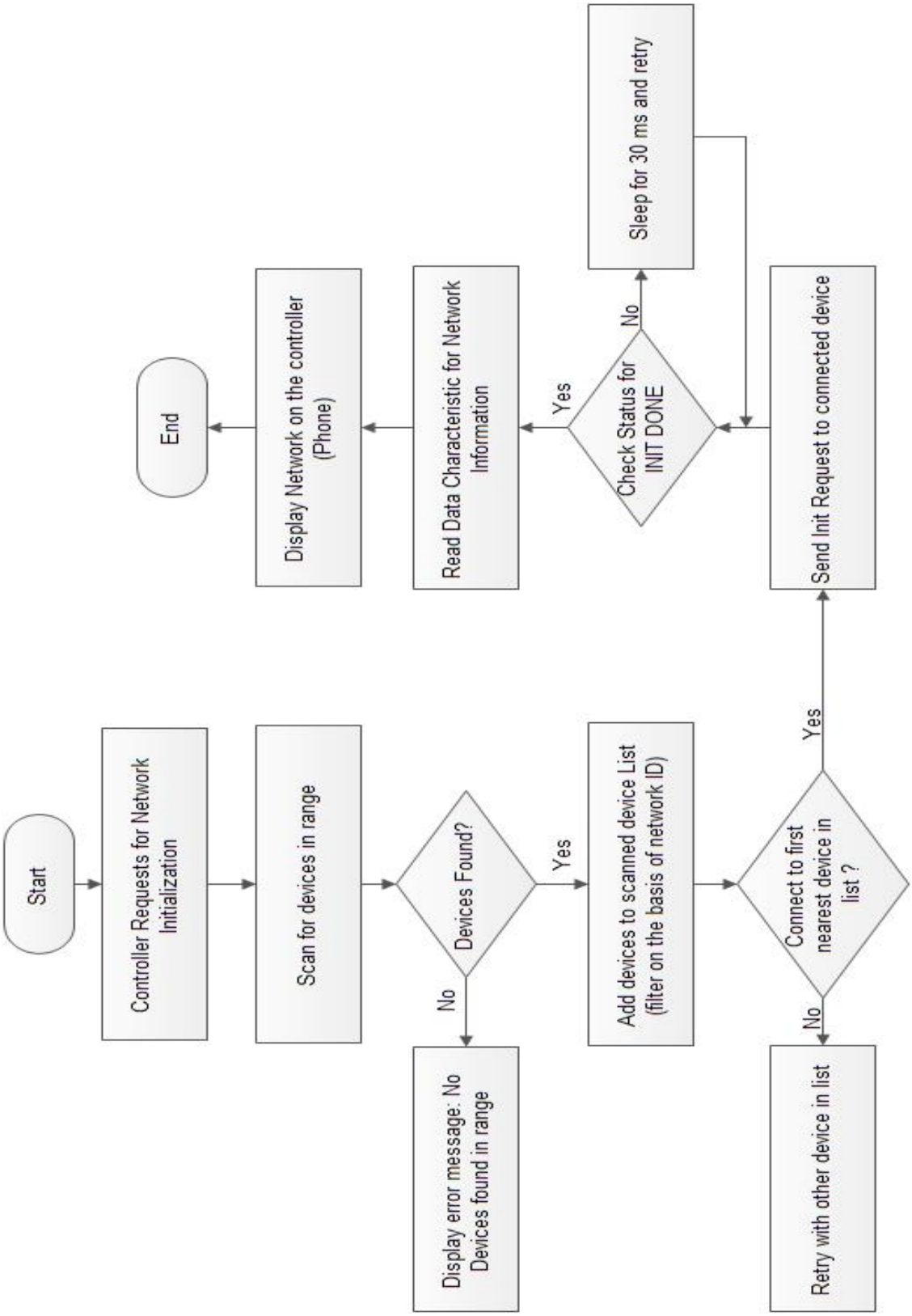

Figure 12 Network Initialization Algorithm (Controller End)

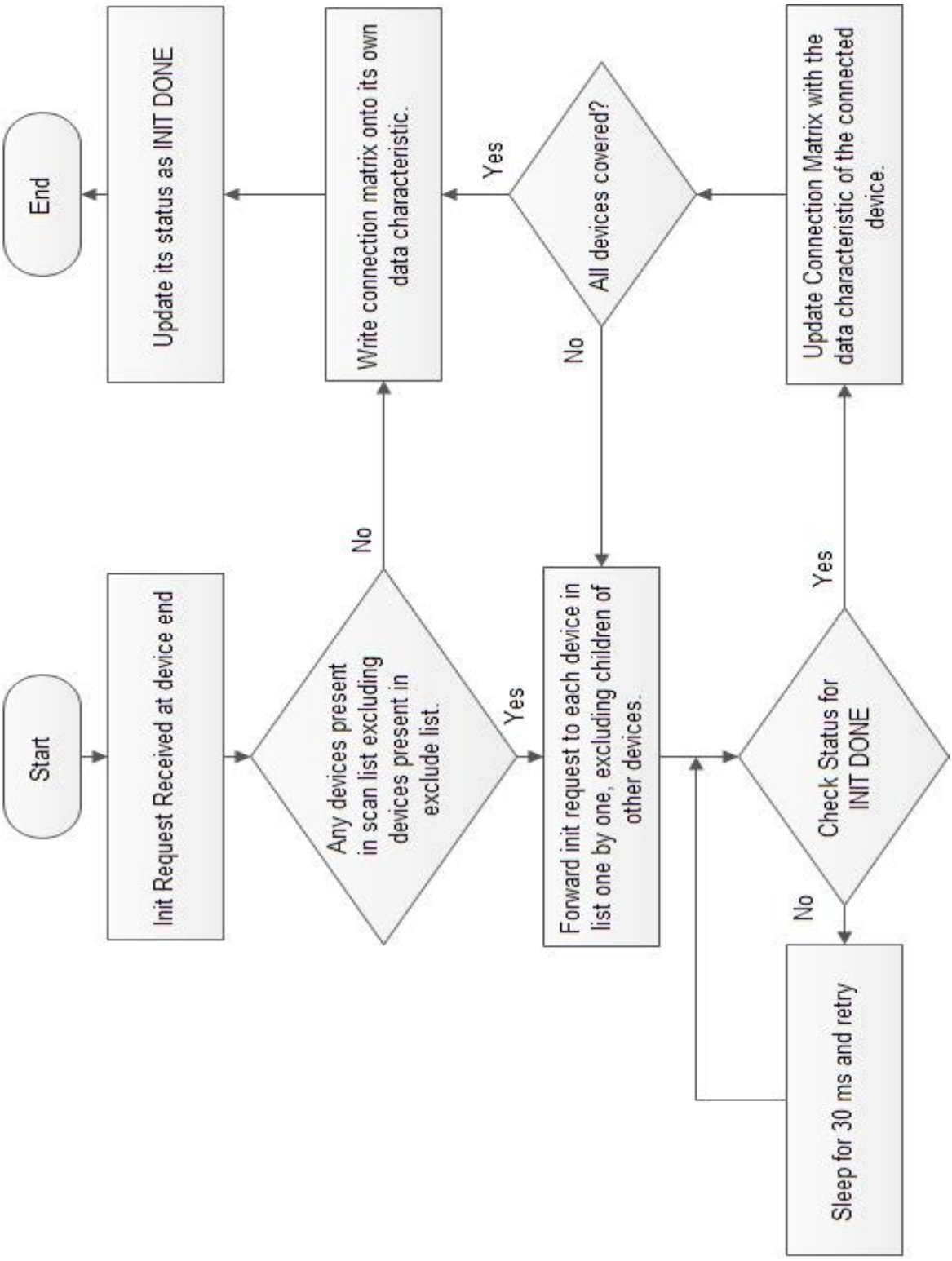

Figure 13: Network Initialization Algorithm (Device End)

4.4 Phase 2: Device Control

To switch on or off a device, the control request needs to be sent from the Phone. If the device is available in proximity of controller, then the control request can be sent directly to device through controller upon connection establishment. But in case the device is not available in proximity of controller, the path to that device needs to be evaluated and accordingly controller sends the request. To achieve this, we can have two ways to carry forward the request assuming the worst case of up-to 64 nodes in a skewed tree structure as in Figure 14.

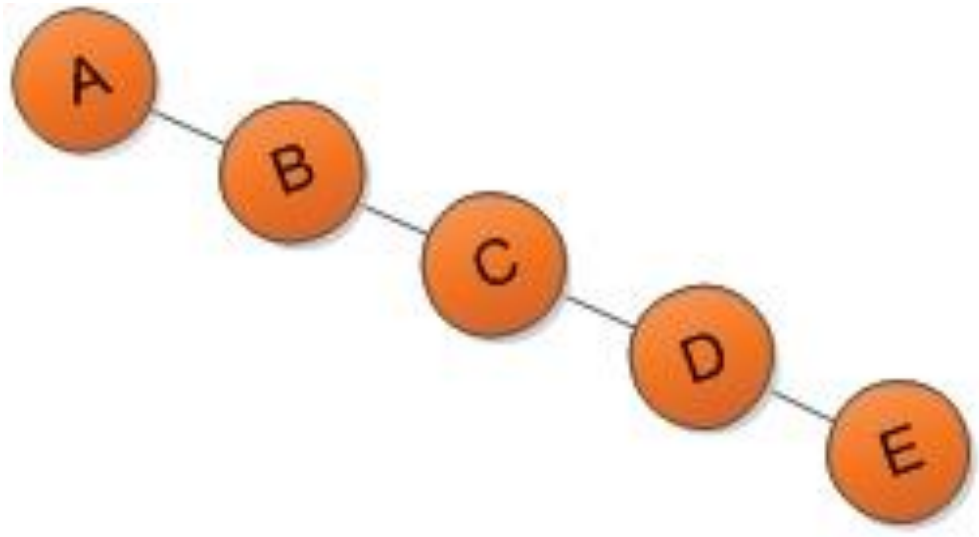

Figure 14: Skewed Tree

Case 1: In this case, complete path is evaluated at the beginning on controller side and that complete information is forwarded at each device. This method reduces the calculations at device level but number of packets to be communicated increases. For example for a network in figure 7, to store path of 64 devices we need 6 packets. So total number of packets add up to

204(6*4+12*5+12*4+12*3+12*2+12*1 = 204)

Case 2: In this case, we evaluate the complete path on controller and forward information of only up to 12 nodes in one packet to next device. At the 12th device, the path is again evaluated up-to next 12 devices and forwarded further and so-on till the last device in the network. In this case the number of packets to be send reduces to

$$64(12*1+12*1+ 12*1 + 12*1 + 12*1 + 4*1 = 64)$$

The proposed algorithm for finding the shortest path to send control command from controller to destination device is discussed in form of flowchart in Figure 15 and Figure 16.

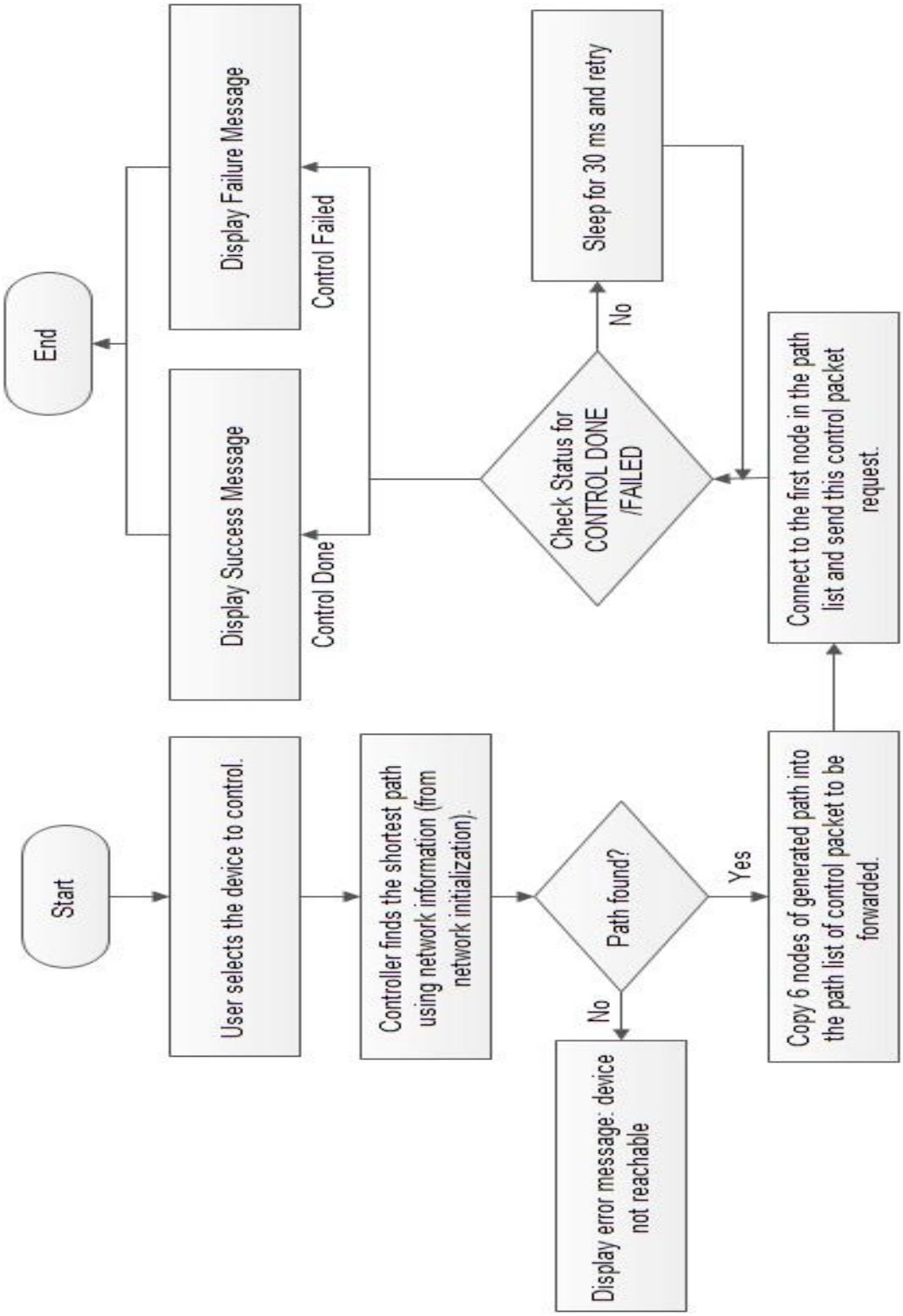

Figure 15: Device control (Controller end)

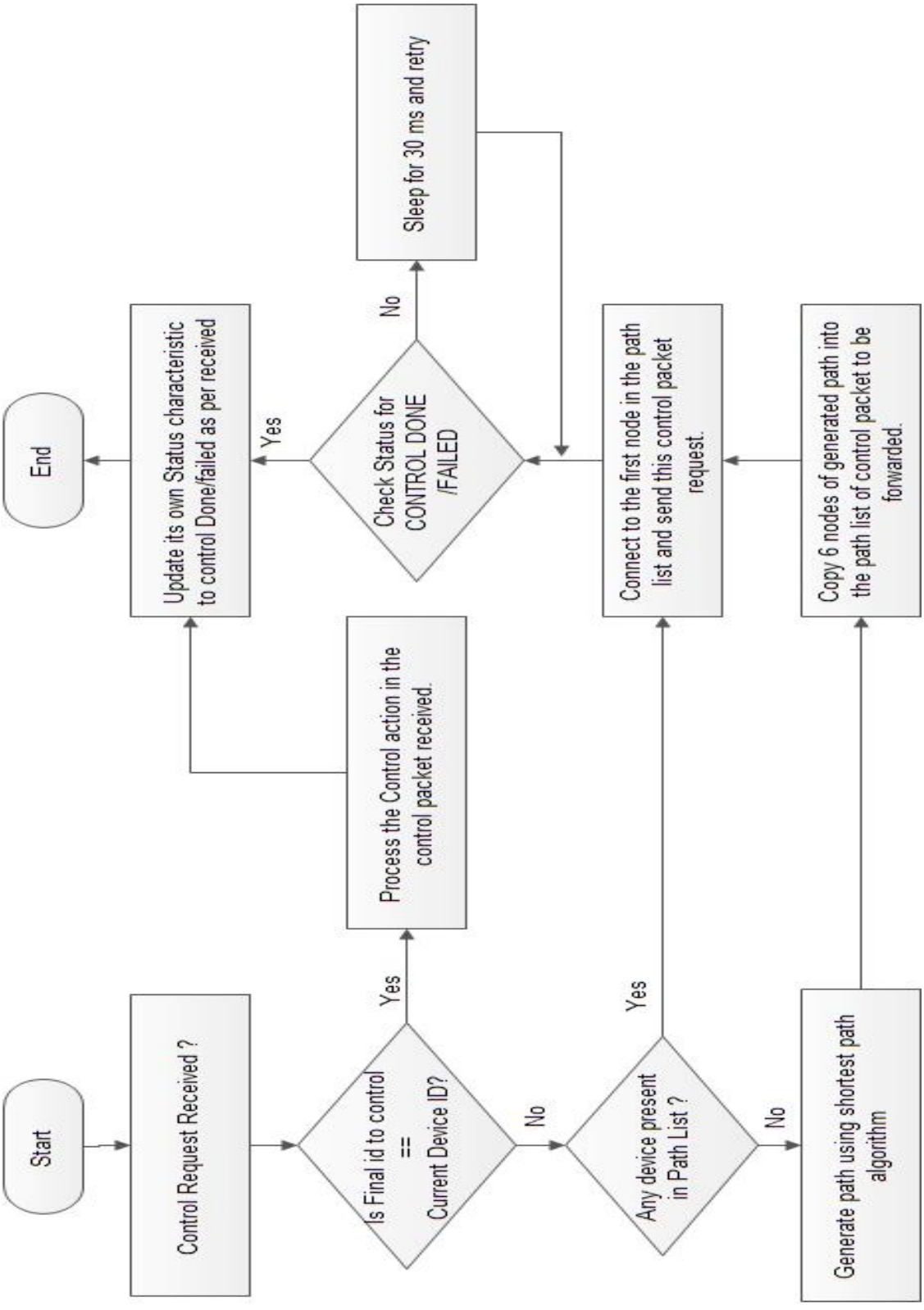

Figure 16: Device Control (Device end)

4.5 Self – Healing Network

In this proposed protocol, the network is considered to be Self- healing in handling the addition of a new node as well as when an existing node in an initialized network becomes unresponsive or dead.

When a new node say X enters an initialized network, X sends request to one of the neighboring nodes, say A and collect all the network information. When A receives the above request, it performs the tasks as per the flowchart:

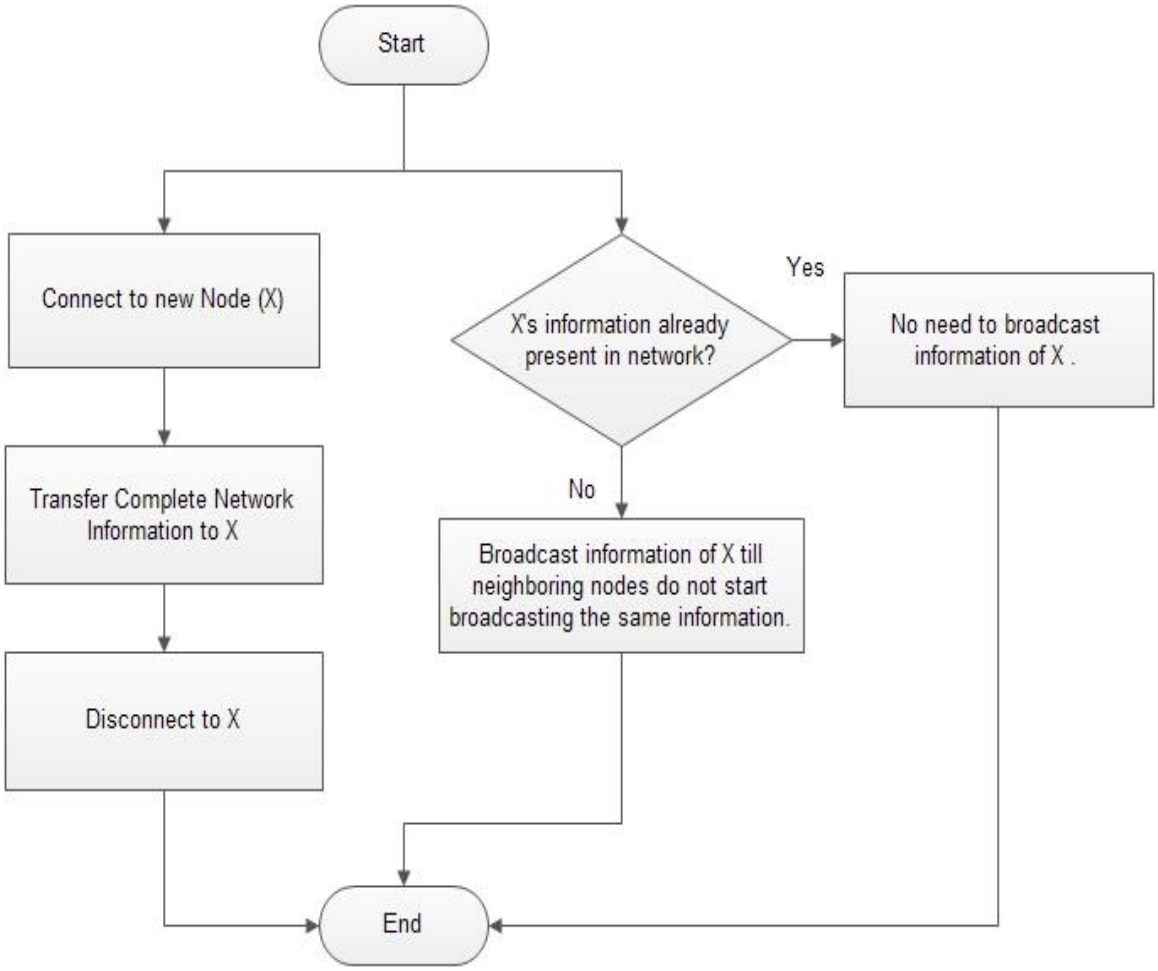

Figure 17: Addition of a new Device X in an Initialized Network

Other nodes will also broadcast similar to A when they receive the information of X.

When an existing node is unresponsive / not reachable,

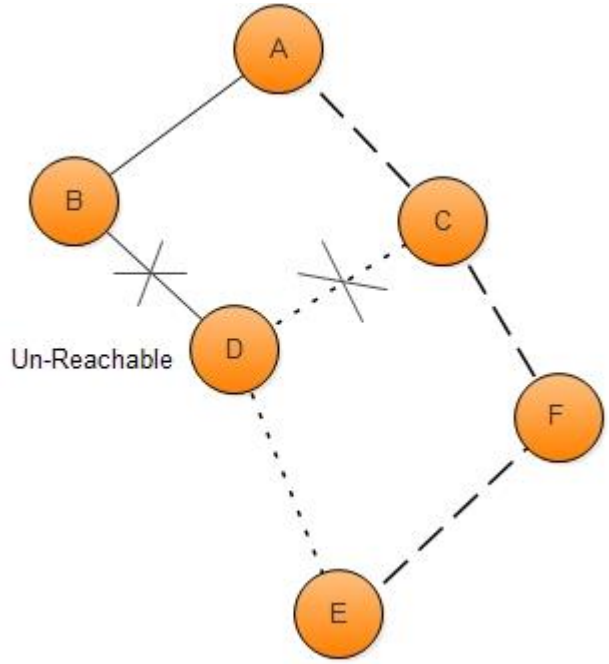

Figure 18: D is the Dead not in Network

- If after a few scans a node say B, doesn't find advertisement packet of its child nodes say D, it can mark the state of D as dead in their connection matrix.
- When Phone controls a device E, whose path goes from (A->B->D->E),
    - New path will be generated at B i.e. A->C->F->E and will be sent as a control status to A and A will further transfer control to E via A->C->F->E.
- If D is the node to be controlled, then the phone will display error message, that node is invalid or un-reachable.

Chapter 5

IMPLEMENTATION RESULTS

5.1 Implementation snapshots

Implemation has been done and tested using 6 devices as shown in Figure 197. The devices have ID 4, 8, 16, 24, 32 and 40 respectively. With 4 in proximity of android phone. For implemenation to be visible properly the power of devices has been reduced to level 2 and devices are allowed to be in proximity with fixed devices like 4 to be range of 8 and 8 to be in range of 4 and 16 and so on as the network is displayed over the phone.

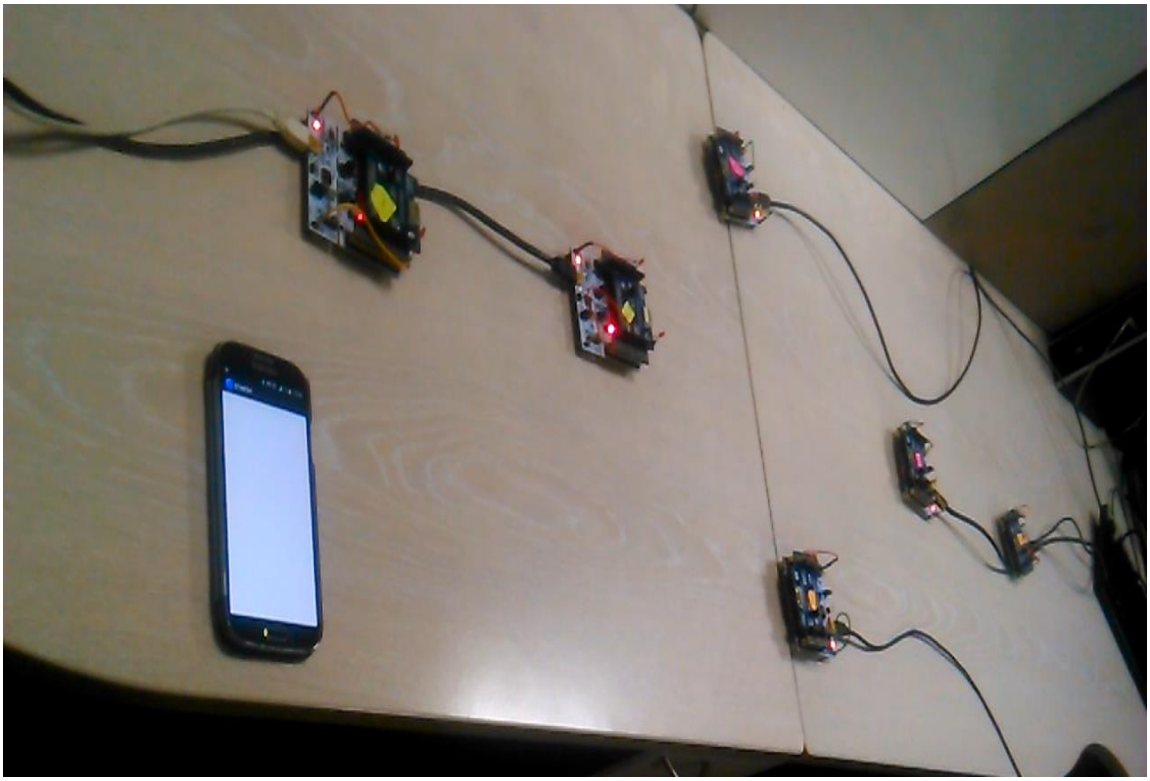

Figure 19: Initial Set up

Implementation involves following steps:

Step 1: Controller (Android Phone) sends request for Network Initialization to nearby device. Figure 208, displays the step 1, controller sends initialization request to nearby device with ID 4 (with Green LED ON signifying that the device has been connected through phone).

Figure 20: Step 1 init request from phone to device

Device 4 already has device 8 in its proximity, so phone can see that device 8 exists in the network and displays it on screen before initialization complete as in Figure 21. The green dot in the Figure 21 denotes the controller (Phone) and lines show that device 4 is in range of Phone and device 8 is in range of device 4.

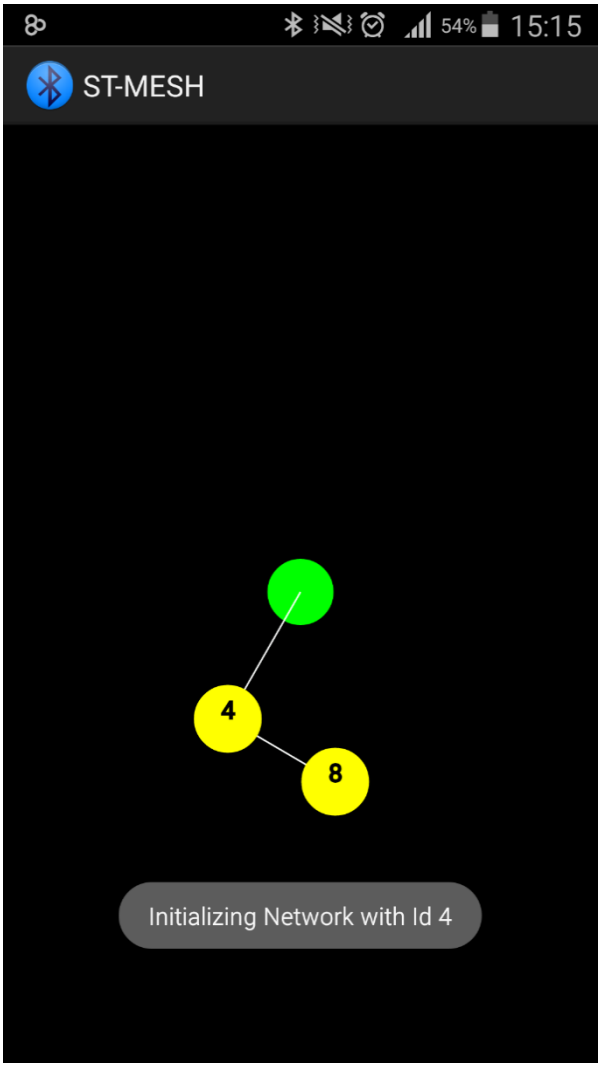

Figure 21: Implementation Step 1

Step 2: Device 4 then starts with Network Initialization Algorithm (Data Collection and Data Distribution). Figure 22 depicts the completion of Step 2 and the green led of device 4 is switched OFF after this step completion because phone disconnects to the device after receiving init done request.

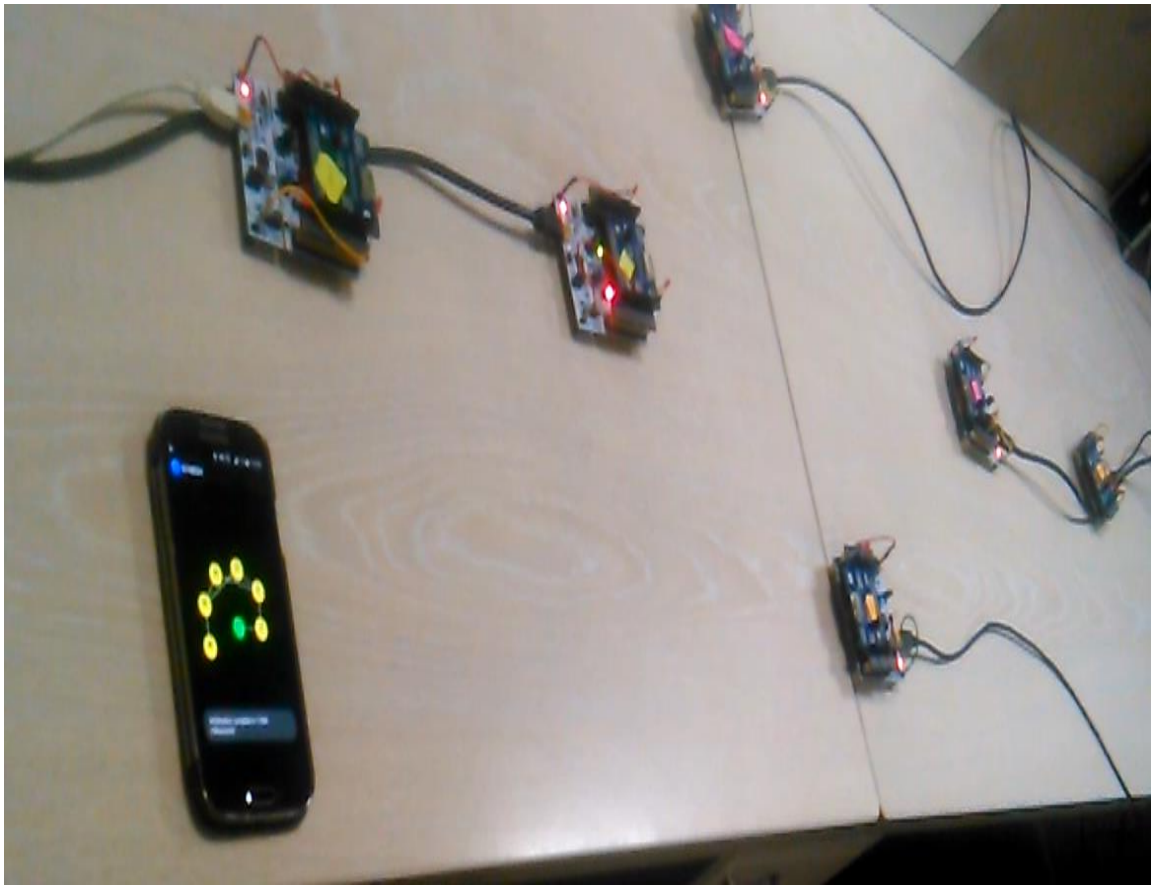

Figure 22: After completion of step 2

Step 3: After Initialization done, controller displays the network as in Figure 23 . The display is refreshed once the initialization done has been received on the controller end.

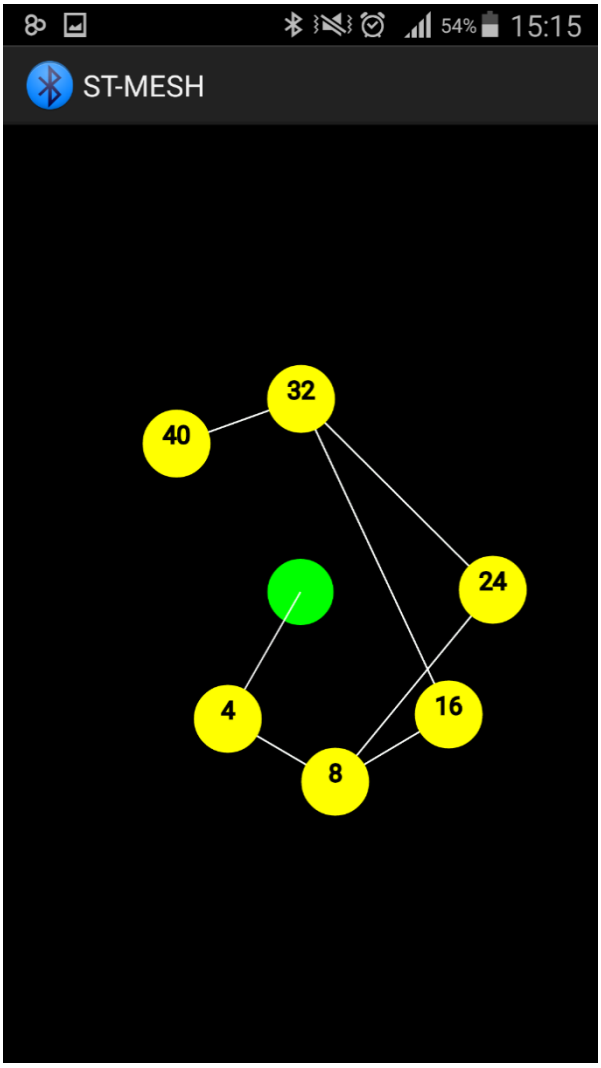

Figure 23: After initialization done (Step 2 and 3)

Step 4: When a particular device is selected from controller to On/Off, a control command is sent to device following device control algorithm. If the Device is on the color of the device changes to grey in the display network as displayed in Figure 24. In Figure 24: Step 4 after devices are switched ON, devices with id 8, 16, 24 and 40 are switched ON by user by clicking on each device dot on the display.

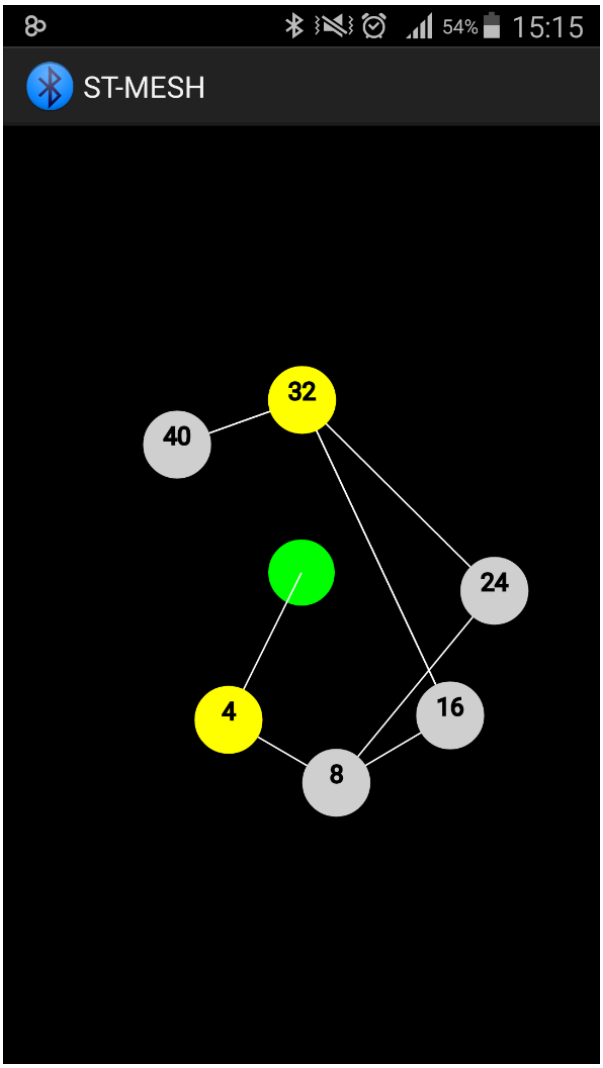

Figure 24: Step 4 after devices are switched ON

5.2 Results

The total number of packets required to be transmitted from source to destination based on number of hops in Case 1 and Case 2 as explained on page 24 is depicted in Figure 25. For large number of hops case 1 will become significantly slow as compared to case 2. In case 2 we may require additional memory and processing which adds some time delay at calculating nodes. Case 2 being the best case, so we implemented the same in the actual implementation.

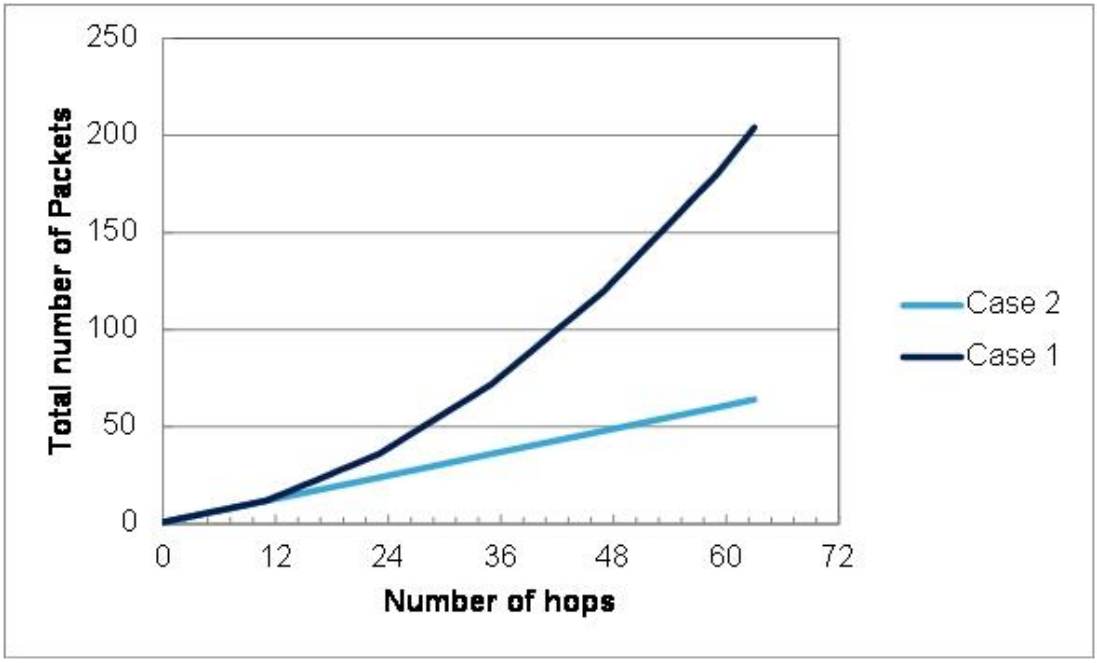

Figure 25: plot of case 1 and case 2 for device control

5.3 Comparison with existing solutions

Considering network topology in Figure 26, a comparison is given in table II. The number of packets consumed for communication from source [A] to destination [E] considering 96 bytes of data transfer in different solutions are given in

Table ***4***.

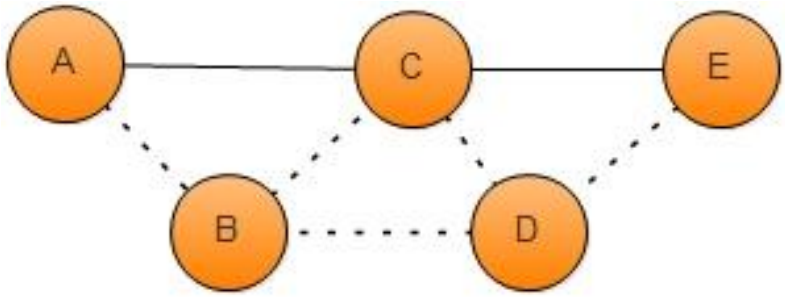

Figure 26: Comparison network

CSRMesh [29] [30] is used to control devices, with no direct communication channel for data transfer and same in case of BleMesh [4]. In proposed method, a fixed shortest path between two nodes is created for data transfer on the basis of initialized network. When connected with each other, devices can transfer 22 bytes of data can be transferred in one transmission. This evaluates to 5 packets for 96 bytes data transfer. Number of transmissions required for transferring 5 packets from A to E reduces to 12 [2*(1 control command packet + 5 data packets)]. Thus providing a faster communication method in lesser transmissions. Also, in flooding techniques. Software level encryption is used which consumes higher CPU resources as compared to hardware level encryption. In our proposed method, connections are maintained and not terminated until required. Connections can be terminated as soon as the task is performed if power saving is required.

Table 4: Comparison with Existing Solutions

Parameter	**CSRMESH[6]**	**BleMesh[3]**	**Proposed solution (ST-Mesh)**
No of packets (96 bytes)	12 (8 bytes payload)	5 (21 bytes payload)	5 (22 bytes payload)
No of Transmissions	96	16	12
No of Connections	0	0	2
Data Transfer Capability	No	No	Yes
Chipset	CSR101X	TI's CC2540 SOC	BlueNRG-MS
Technique	Flooding	Opportunistic Routing	Connection Based Mesh Formation
Security	Software encryption	-	Hardware encryption

Chapter 6

CONCLUSION

In this dissertation report, a reliable solution for formation of mesh network for home automation using BlueNRG-MS devices has been proposed and implemented. In the presented approach, one time initialization may take some time, but the controlling of devices is comparatively fast and more reliable. Also, network is self-healing when an existing node is removed from network as well as in case when a device is not able to communicate to other device via an existing path. Results show that number of packets communicated is reduced as compared to flooding technique and opportunistic routing used in existing solutions.

This proposed algorithm can further be optimized by finding a minimum spanning tree for data distribution. Furthermore, we will consider energy consumption of devices while performing computation to make our approach more optimized and cost effective.

Chapter 7

REFERENCES

[1] J. Decuir, "Introducing Bluetooth Smart Part 1: A look at both classic and new technologies," *IEEE Consumer Electronics Magazine,* 2014.

[2] K.-H. CHANG and C. CONSULTING, "BLUETOOTH:A VIABLE SOLUTION FOR IOT?," *IEEE Wireless Communications,* 2014.

[3] J. Decuir, "Introducing Bluetooth Smart Part II: Applications and updates," *IEEE Consumer Electronics Magazine,* 2014.

[4] H. Kim, J. Lee and J. Jang, " BLEmesh: A wireless mesh network protocol for bluetooth low energy devices," in *2015 3rd International conference on future Internet of Things and Cloud,*, 2015.

[5] M. Siekkinen, M. Hiienkari, J. K. Nurminen and and J. Nieminen, " How Low is Bluetooth Low Energy? Compartive Mesasurements with Zigbee/ 802.15.4," in *2012 IEEE Wireless Communications and Networking Converence Workshops (WCNCW), April. 2012*, 2012.

[6] *Bluetooth Special Interest Group, Bluetooth Core Specification Version 4.2, December, 2014*..

[7] K. Townsend, C. Cufí, Akiba and R. Davidson, Getting Started with Bluetooth Low Energy.

[8] "Online Resource: http://developer.android.com/guide/topics/connectivity/bluetooth.html,"

[9] "http://www.st.com/web/catalog/tools/FM146/CL2167/SC2006/PF262191," [Online].

[10] "http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1847/PF260002?icmp=nucleo-ipf_pron_pr-nucleo_feb2014&sc=nucleoL152RE-pr," [Online].

[11] E. Mackensen, M. Lai and T. M. Wendt, "Performance Analysis of an Bluetooth Low Energy Sensor System," in *The 1st IEEE International Symposium on Wireless*

Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems, Germany, 2012.

[12] E. Mackensen, M. Lai and T. M. Wendt, "Bluetooth Low Energy (BLE) based wireless sensors," in *Sensors, 2012 IEEE*, Taipei, 2012.

[13] A. Dementyev, S. Hodges, S. Taylor and J. Smith, "Power Consumption Analysis of Bluetooth Low Energy, ZigBee and ANT Sensor Nodes in a Cyclic Sleep Scenario," in *Wireless Symposium (IWS), 2013 IEEE International*, Beijing, 2013.

[14] J. Liu, C. Chen, Y. Ma and Y. Xu, "Energy Analysis of Device Discovery for Bluetooth Low Energy," in *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*, Las Vegas, NV, 2013.

[15] X. Zhao, Z. Xiao, A. Markham, N. Trigoni and Y. Ren, "Does BTLE measure up against WiFi? A comparison of indoor location performance," in *European Wireless 2014*, 2014.

[16] K. Shahzad and B. Oelmann, "A Comparative Study of In-sensor Processing vs Raw Data Transmission using ZigBee, BLE and Wi-Fi for Data Intensive Monitoring Applications," in *Wireless Communications Systems (ISWCS), 2014 11th International Symposium*, 2014.

[17] K. Mikhaylov, "Simulation of Network-Level Performance for Bluetooth Low Energy," in *2014 IEEE 25th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2014.

[18] H. Lee, D. Ok, J. Han, I. Hwang and K. Kim, "Performance Anomaly of Neighbor Discovery in Bluetooth Low Energy," in *2016 IEEE International Conference on Consumer Electronics (ICCE)*, 2016.

[19] J.-S. Lee, M.-F. Dong and Y.-H. Sun, "A Preliminary Study of Low Power Wireless Technologies: ZigBee and Bluetooth Low Energy," in *Industrial Electronics and Applications (ICIEA), 2015 IEEE 10th Conference*, 2015.

[20] P. D. Marco, R. Chirikov, P. Amin and F. Militano, "Coverage Analysis of Bluetooth Low Energy and IEEE 802.11ah for Office Scenario," in *2015 IEEE 26th International Symposium on Personal, Indoor and Mobile Radio Communications -*

(PIMRC): Workshop on, 2015.

[21] R. Frank, W. Bronzi, G. Castignani and T. Engel, "Bluetooth Low Energy: An Alternative Technology for VANET Applications," in *Wireless On-demand Network Systems and Services (WONS), 2014 11th Annual Conference*, 2014.

[22] J.-R. Lin, T. Talty and O. K. Tonguz, "A Blind Zone Alert System Based on Intra-Vehicular Wireless Sensor Networks," in *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 11, NO. 2, APRIL 2015*, 2015.

[23] B.-G. Lee, B.-L. Lee and W.-Y. Chung, "Wristband-Type Driver Vigilance Monitoring System Using Smartwatch," *IEEE SENSORS JOURNAL, VOL. 15, NO. 10, OCTOBER 2015*, 2015.

[24] Y. S. Cho, J. Kwon and S. Choi, "Development of Smart LED Lighting System Using Multi–Sensor Module and Bluetooth Low Energy Technology," in *2014 IEEE SECON Posters -- IEEE International Conference on Sensing, Communications and Networking (SECON)*, 2014.

[25] M. Grover, D. S. K. Pardeshi, N. Singh and S. Kumar, "BLUETOOTH LOW ENERGY FOR INDUSTRIAL AUTOMATION," in *IEEE SPONSORED 2ND INTERNATIONAL CONFERENCE ON ELECTRONICS AND COMMUNICATION SYSTEM*, 2015.

[26] M. Choi, W.-K. Park and I. Lee, "Smart Office Energy Management System Using Bluetooth Low Energy Based Beacons and a Mobile App," in *2015 IEEE International Conference on Consumer Electronics (ICCE)*, 2015.

[27] M.-J. Su, H.-S. Chen, Y.-J. Lin, I.-L. Chen and C.-T. Cheng, "Continuous Spine Care Service for Elderly," in *2015 Eighth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, 2015.

[28] Y. Ding, S. Gang and J. Hong, "The Design of Home Monitoring System by Remote Mobile Medical," in *2015 7th International Conference on Information Technology in Medicine and Education*, 2015.

[29] "Online Resource: http://wiki.csr.com/wiki/CSRmesh.," [Online].

[30] "Online Resource: http://www.csr.com/blog/2015/04/csrmesh-faqs-2/. .," [Online].

[31] C. M. Y. a. J. -H. Lin, "Enhanced Bluetree: a mesh topology approach forming Bluetooth scatternet," *2012 IET Wireless Sensor Systems, 2012*, pp. 409-415.

[32] Z. Guo, I. G. Harris, L. Tsaur and X. Chen, "An on-demand Scatternet Formation and Multi-hop Routing Protocol for BLE-based wireless sensor networks," in *2015 IEEE Wireless Communications and Networking Conference,*, 2015.

[33] "Online Resource: https://github.com/mwaylabs/fruitymesh/wiki.," [Online].

[34] "OnlineResource: http://blog.mwaysolutions.com/2015/08/28/bluerangemeshing-now-open-source/," [Online].