LEACH in WSN

Software: NetSim Standard v14.3, Visual Studio 2022

Project Download Link:

https://github.com/NetSim-TETCOS/LEACH-in-WSN-v14.3/archive/refs/heads/main.zip

Follow the instructions specified in the following link to download and set up the Project in NetSim:

https://support.tetcos.com/en/support/solutions/articles/14000128666-downloading-and-setting-up-netsim-fileexchange-projects

1 Introduction

Low-energy adaptive clustering hierarchy ("LEACH") is a MAC protocol that is integrated with clustering and a simple routing protocol in wireless sensor networks (WSNs). The goal of LEACH is to lower the energy consumption required to create and maintain clusters to improve the lifetime of a wireless sensor network.

This Cross-Layer Protocol is implemented in NetSim in the MAC layer which involves ZigBee Protocol and the Network layer which involves DSR protocol. The clustering of sensors happens in the Network layer and the Cluster head election involves interacting with the MAC layer to obtain the remaining power of the sensors.

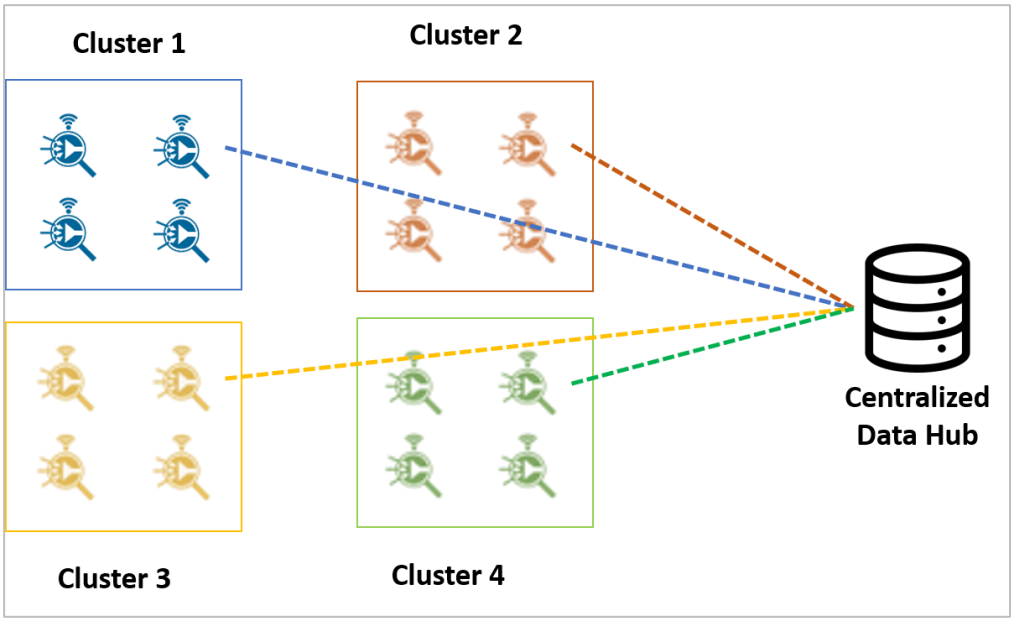


Figure 1: Cluster-based node organization in WSN

2 Implementation of Leach in WSN

A **Leach .c** file is added to the DSR project.

1. For this implementation of Leach, the number of Clusters is fixed as 4 and all the 4 clusters are equal. If the user wants to change it, then he/she must also change the static routing for the Cluster Heads and the ClusterElement array accordingly in **Leach.c**

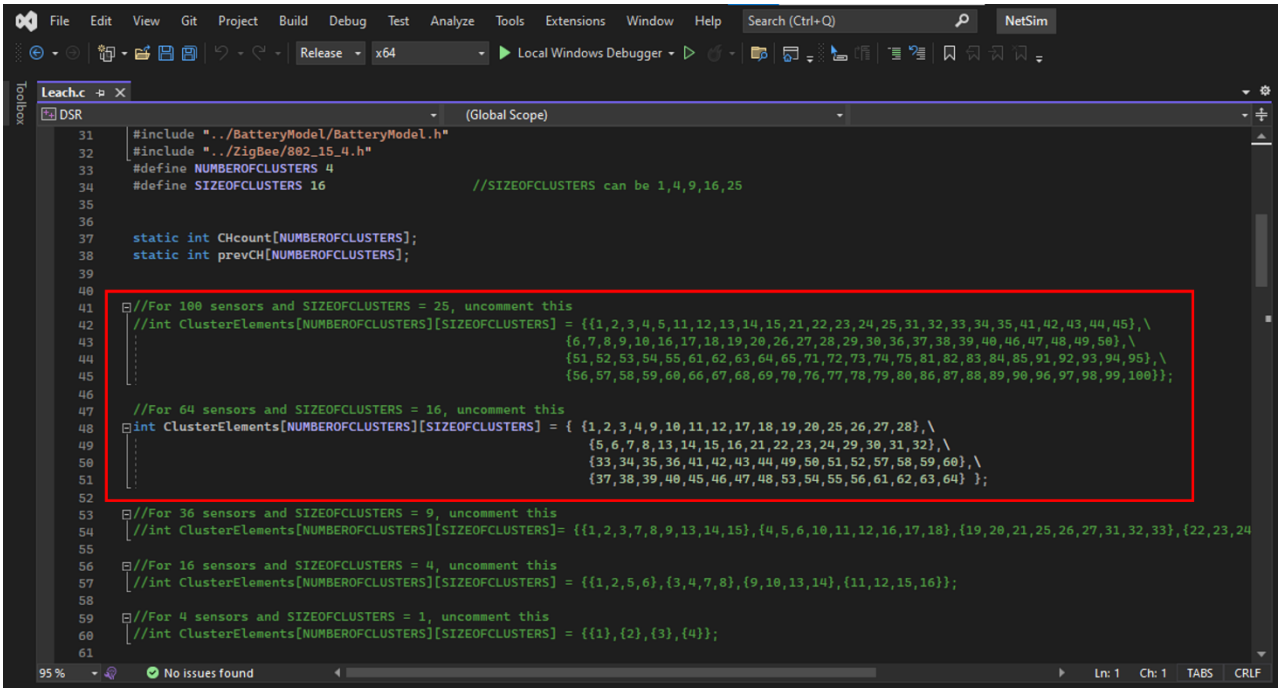


Figure 2: Leach.c file

2. To make 4 equal clusters the number of sensors must be 4,16,36,64,100. Depending on the number of sensors, the Cluster Elements array must be defined. Here, it has been defined and commented for 4,16,36,64,100 sensors. Uncomment the one you want to use.

The file contains the following functions:

fn_NetSim_LEACH_CheckDestination(); //This function is used to check whether the current device is the destination (i.e) the sink node or not. Else the packet will be forwarded to the next hop.

fn_NetSim_LEACH_GetNextHop(); //This function is used to identify the next hop in cases where the current device is either a sensor within the cluster or the cluster head. Static routes are defined in this function. It returns the Device id of the next hop.

fn_NetSim_LEACH_AssignClusterHead (); //This function is used to dynamically assign cluster heads within a cluster based on the residual energy. The sensor with higher remaining power in comparison to other sensors within the same cluster will be elected as the cluster head.

fn_NetSim_LEACH_IdentifyCluster(); //This function is used to determine the cluster to which a sensor belongs. It returns the cluster id of the cluster.

3 Example

1. The **LEACH-in-WSN-Workspace** comes with a sample network configuration that is already saved. To open this example, go to Your work in the Home screen of NetSim and click on the **LEACH-in-WSN-Example**. from the list of experiments.

2. The network scenario consists of 64 sensors uniformly placed along with the SINKNODE as shown below.

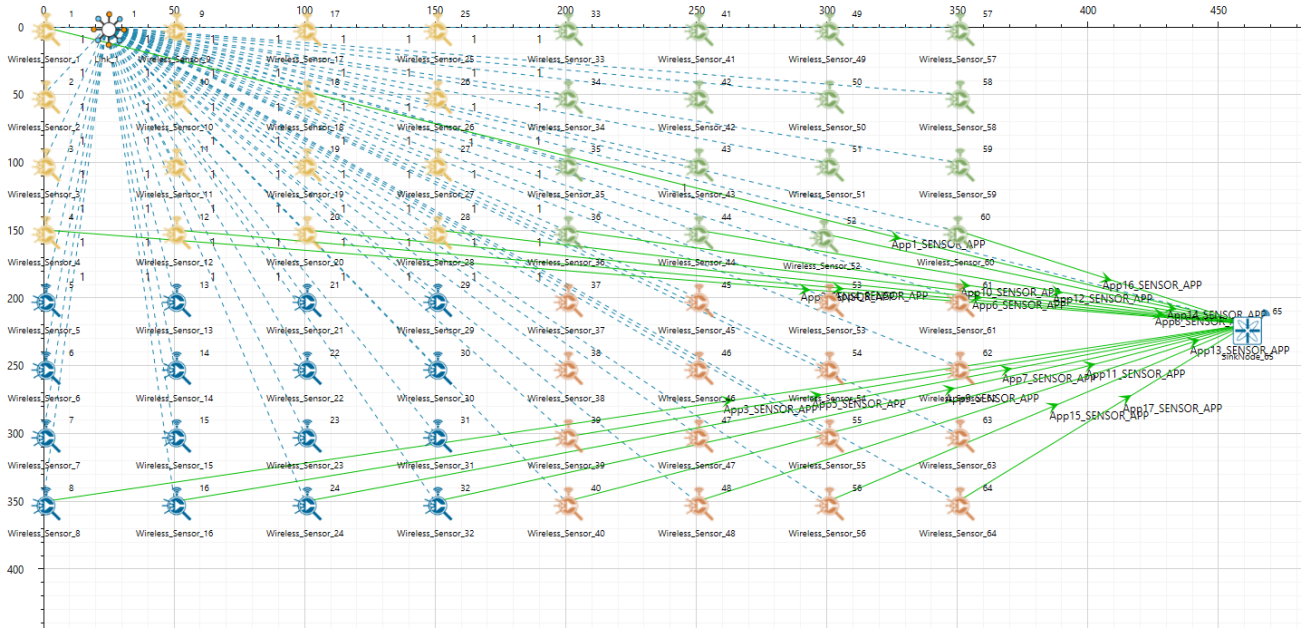


Figure 3: Network topology in the project

3. Run the simulation.

4 Results and discussion

In packet trace, you will notice that the sensors directly start transmitting packets without route establishment since the routes are statically defined in LEACH.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
1	0	Sensing	App10_SENSOR_APP	SENSOR-36	SINKNODE-65	SENSOR-36	SENSOR-49
1	0	Sensing	App17_SENSOR_APP	SENSOR-64	SINKNODE-65	SENSOR-64	SENSOR-53
1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-19
1	0	Sensing	App14_SENSOR_APP	SENSOR-52	SINKNODE-65	SENSOR-52	SENSOR-49
1	0	Sensing	App8_SENSOR_APP	SENSOR-28	SINKNODE-65	SENSOR-28	SENSOR-17
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-17	SENSOR-28	SENSOR-17	SENSOR-28
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-4	SENSOR-17
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-17	SENSOR-4	SENSOR-17	SENSOR-4
1	0	Sensing	App12_SENSOR_APP	SENSOR-44	SINKNODE-65	SENSOR-44	SENSOR-49
1	0	Sensing	App4_SENSOR_APP	SENSOR-12	SINKNODE-65	SENSOR-12	SENSOR-19
1	0	Sensing	App5_SENSOR_APP	SENSOR-16	SINKNODE-65	SENSOR-16	SENSOR-22
1	0	Sensing	App10_SENSOR_APP	SENSOR-36	SINKNODE-65	SENSOR-36	SENSOR-49
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-49	SENSOR-36	SENSOR-49	SENSOR-36
1	0	Sensing	App5_SENSOR_APP	SENSOR-16	SINKNODE-65	SENSOR-16	SENSOR-22
1	0	Sensing	App10_SENSOR_APP	SENSOR-36	SINKNODE-65	SENSOR-49	SINKNODE-65
0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-65	SENSOR-49	SINKNODE-65	SENSOR-49
1	0	Sensing	App12_SENSOR_APP	SENSOR-44	SINKNODE-65	SENSOR-44	SENSOR-49
1	0	Sensing	App4_SENSOR_APP	SENSOR-12	SINKNODE-65	SENSOR-12	SENSOR-19
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-4	SENSOR-17
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-17	SENSOR-4	SENSOR-17	SENSOR-4
1	0	Sensing	App8_SENSOR_APP	SENSOR-28	SINKNODE-65	SENSOR-17	SENSOR-49
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-49	SENSOR-17	SENSOR-49	SENSOR-17
1	0	Sensing	App12_SENSOR_APP	SENSOR-44	SINKNODE-65	SENSOR-44	SENSOR-49
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-49	SENSOR-44	SENSOR-49	SENSOR-44
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-17	SENSOR-49
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-49	SENSOR-17	SENSOR-49	SENSOR-17
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-17	SENSOR-33
1	0	Sensing	App8_SENSOR_APP	SENSOR-28	SINKNODE-65	SENSOR-49	SINKNODE-65
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-17	SENSOR-33
0	N/A	Control_Packet	Zigbee_ACK	SENSOR-33	SENSOR-17	SENSOR-33	SENSOR-17
1	0	Sensing	App2_SENSOR_APP	SENSOR-4	SINKNODE-65	SENSOR-33	SINKNODE-65
1	0	Sensing	App8_SENSOR_APP	SENSOR-28	SINKNODE-65	SENSOR-49	SINKNODE-65
0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-65	SENSOR-49	SINKNODE-65	SENSOR-49
1	0	Sensing	App12_SENSOR_APP	SENSOR-44	SINKNODE-65	SENSOR-49	SINKNODE-65

Figure 4: NetSim Packet trace results for packet transmission

The cluster heads change dynamically. Users can observe the cluster head selection by filtering an application in the packet trace, as shown in the figure below.

Additionally, by filtering the transmitter ID, users can identify the node or sensor that is elected as the cluster head for a particular sensor.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-19
1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-19
2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-2
2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-2
2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-2
5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-10
5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-10
5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-10
5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-10
10	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-27
10	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-27
10	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-27
10	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-27
10	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-65	SENSOR-1	SENSOR-27

Figure 5: NetSim Packet trace results for cluster head selection

The battery model table in the Result Dashboard window reveals that the consumed energy is significantly lower with LEACH protocol implementation compared to without LEACH Protocol.This

can be observed in the battery model table by clicking the additional metrics on the left in Results dashboard window.

With Leach Protocol Implementation:

Battery model

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Harvested Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	1189.883912	3886952.746449	142.630361
WIRELESS_SENSOR_2	3888000.000000	1182.335571	3886960.294790	142.630361
WIRELESS_SENSOR_3	3888000.000000	1183.989633	3886958.640727	142.630361
WIRELESS_SENSOR_4	3888000.000000	1190.946850	3886951.683511	142.630361
WIRELESS_SENSOR_5	3888000.000000	1180.039831	3886962.590529	142.630361
WIRELESS_SENSOR_6	3888000.000000	1181.760450	3886960.869911	142.630361
WIRELESS_SENSOR_7	3888000.000000	1181.126268	3886961.504093	142.630361

Figure 6: Battery model table

Without Leach Protocol Implementation:

Without LEACH battery model results can be obtained by resetting the binaries option present under your work in NetSim home screen window.

Battery model

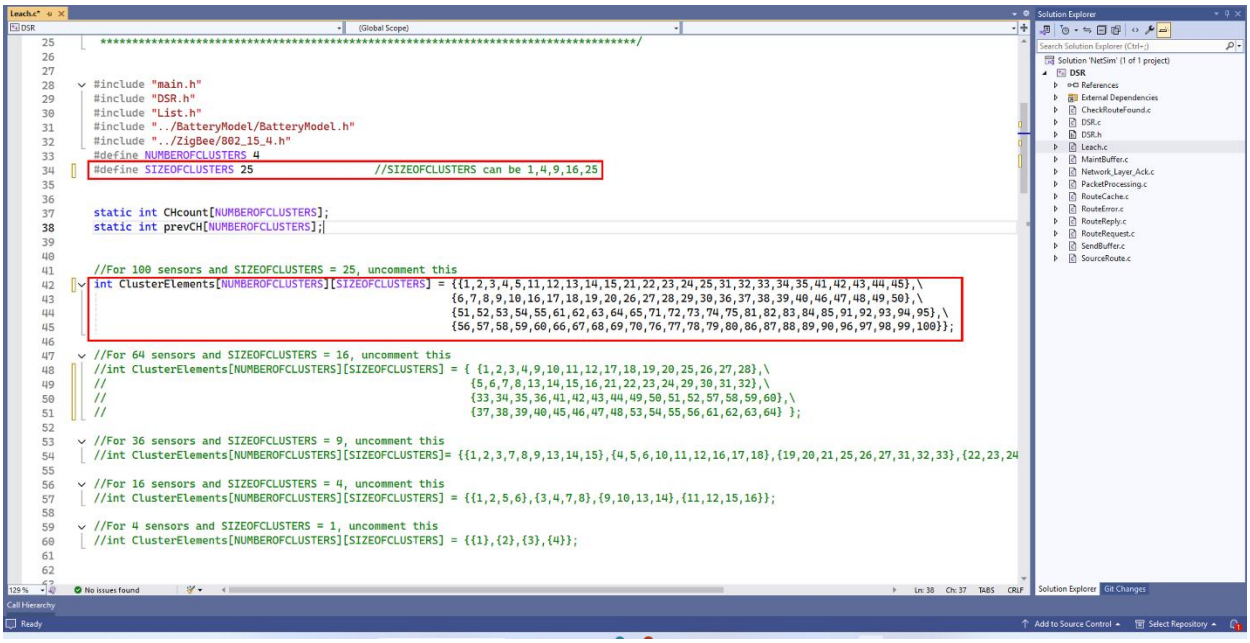
Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Harvested Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	1213.748661	3886929.577444	143.326104
WIRELESS_SENSOR_2	3888000.000000	1196.283876	3886947.042228	143.326104
WIRELESS_SENSOR_3	3888000.000000	1196.283876	3886947.042228	143.326104
WIRELESS_SENSOR_4	3888000.000000	1196.874433	3886946.451671	143.326104
WIRELESS_SENSOR_5	3888000.000000	1196.289682	3886947.036422	143.326104
WIRELESS_SENSOR_6	3888000.000000	1196.286779	3886947.039325	143.326104
WIRELESS_SENSOR_7	3888000.000000	1196.289682	3886947.036422	143.326104

Figure 7: Battery model table

Note: You can observe slight variation in the Consumed energy with and without Leach protocol implementation.

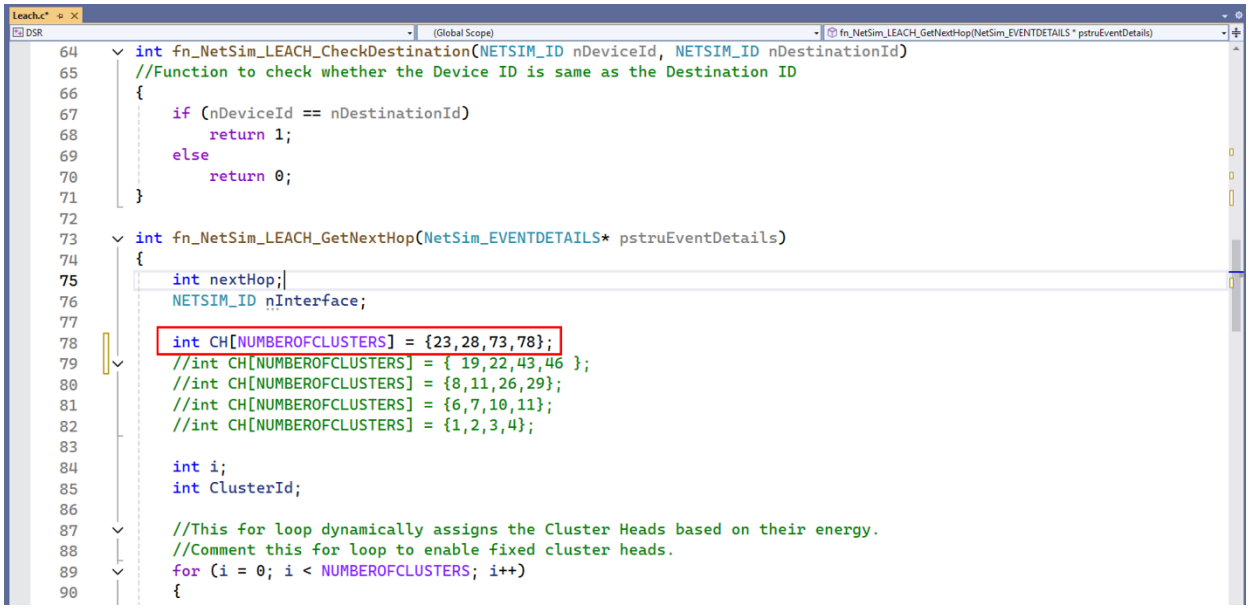
Users can modify the number of nodes by uncommenting the following lines in the Leach.c file within the DSR project.
This allows testing with different sets of sensors, such as 4, 9, or 25 nodes.

Example for 25 sensors,



```
#include "main.h"
#include "DSR.h"
#include "List.h"
#include "../BatteryModel/BatteryModel.h"
#include "../ZigBee/802_15_4.h"
#define NUMBEROFCLUSTERS 4
#define SIZEOFCLUSTERS 25                    //SIZEOFCLUSTERS can be 1,4,9,16,25

static int CHcount[NUMBEROFCLUSTERS];
static int prevCH[NUMBEROFCLUSTERS];

//For 100 sensors and SIZEOFCLUSTERS = 25, uncomment this
int ClusterElements[NUMBEROFCLUSTERS][SIZEOFCLUSTERS] = {{1,2,3,4,5,11,12,13,14,15,21,22,23,24,25,31,32,33,34,35,41,42,43,44,45},\
                                                         {6,7,8,9,10,16,17,18,19,20,26,27,28,29,30,36,37,38,39,40,46,47,48,49,50},\
                                                         {51,52,53,54,55,61,62,63,64,65,71,72,73,74,75,81,82,83,84,85,91,92,93,94,95},\
                                                         {56,57,58,59,60,66,67,68,69,70,76,77,78,79,80,86,87,88,89,90,96,97,98,99,100}};

//For 64 sensors and SIZEOFCLUSTERS = 16, uncomment this
//int ClusterElements[NUMBEROFCLUSTERS][SIZEOFCLUSTERS] = { {1,2,3,4,9,10,11,12,17,18,19,20,25,26,27,28},\
//                                                          {5,6,7,8,13,14,15,16,21,22,23,24,29,30,31,32},\
//                                                          {33,34,35,36,41,42,43,44,49,50,51,52,57,58,59,60},\
//                                                          {37,38,39,40,45,46,47,48,53,54,55,56,61,62,63,64} };

//For 36 sensors and SIZEOFCLUSTERS = 9, uncomment this
//int ClusterElements[NUMBEROFCLUSTERS][SIZEOFCLUSTERS]= {{1,2,3,7,8,9,13,14,15},{4,5,6,10,11,12,16,17,18},{19,20,21,25,26,27,31,32,33},{22,23,24

//For 16 sensors and SIZEOFCLUSTERS = 4, uncomment this
//int ClusterElements[NUMBEROFCLUSTERS][SIZEOFCLUSTERS] = {{1,2,5,6},{3,4,7,8},{9,10,13,14},{11,12,15,16}};

//For 4 sensors and SIZEOFCLUSTERS = 1, uncomment this
//int ClusterElements[NUMBEROFCLUSTERS][SIZEOFCLUSTERS] = {{1},{2},{3},{4}};
```

Along with the following line in function fn_NetSim_LEACH_GetNextHop()



```
int fn_NetSim_LEACH_CheckDestination(NETSIM_ID nDeviceId, NETSIM_ID nDestinationId)
//Function to check whether the Device ID is same as the Destination ID
{
    if (nDeviceId == nDestinationId)
        return 1;
    else
        return 0;
}

int fn_NetSim_LEACH_GetNextHop(NetSim_EVENTDETAILS* pstruEventDetails)
{
    int nextHop;
    NETSIM_ID nInterface;

    int CH[NUMBEROFCLUSTERS] = {23,28,73,78};
    //int CH[NUMBEROFCLUSTERS] = { 19,22,43,46 };
    //int CH[NUMBEROFCLUSTERS] = {8,11,26,29};
    //int CH[NUMBEROFCLUSTERS] = {6,7,10,11};
    //int CH[NUMBEROFCLUSTERS] = {1,2,3,4};

    int i;
    int ClusterId;

    //This for loop dynamically assigns the Cluster Heads based on their energy.
    //Comment this for loop to enable fixed cluster heads.
    for (i = 0; i < NUMBEROFCLUSTERS; i++)
    {
```