TLEACH in WSN

Software: NetSim Standard v14.0, Visual Studio 2022

Project Download Link:

https://github.com/NetSim-TETCOS/TLEACH-in-WSN-v14.0/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-file-exchange-projects

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.

TLEACH is a Threshold-based LEACH in which nodes with remaining energy greater than a threshold value are considered to elect the cluster head. The sensor that is closer to the sink node is elected as the cluster head. Whenever all sensors in a cluster reach an energy level lower than the threshold, the threshold is reduced.

Real-World Context:

In the context of Precision Agriculture in large farms, various monitoring sensors are deployed to gather data on different aspects of the agricultural environment, such as soil moisture, temperature, Weather and crop health. In this example, we consider the TLEACH protocol. T-LEACH (Threshold-sensitive LEACH) is an energy-efficient clustering protocol for wireless sensor networks (WSNs) that is particularly well-suited for precision agriculture applications in large farms. It is an improvement over the original LEACH protocol. In this scenario, we consider Yield,Weather,Soil and Plant as clusters, representing different aspects of the farm's operations. By implementing the TLEACH protocol, we can effectively improve the lifetime of sensors and enhance the efficiency of data collection for precision agriculture in large farms.

Soil Monitoring Cluster: Gathers data on soil moisture, nutrient levels, and temperature.
Plant Monitoring Cluster: Collects data on plant growth and water stress.
Weather Monitoring Cluster: Measures temperature, humidity, precipitation, and wind .
Yield Monitoring Cluster: Estimates crop yield and improve agricultural productivity.

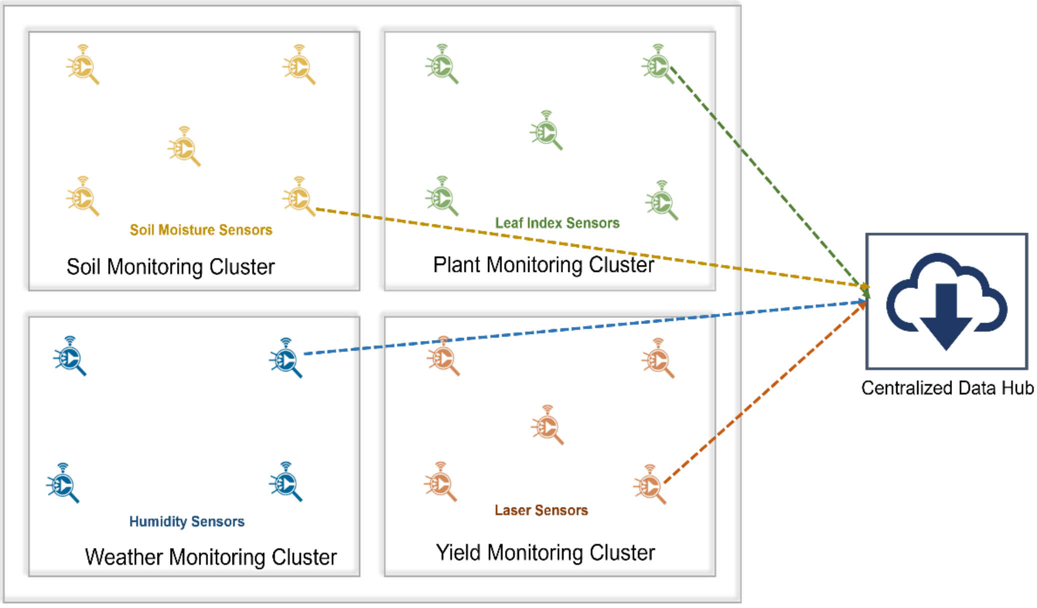
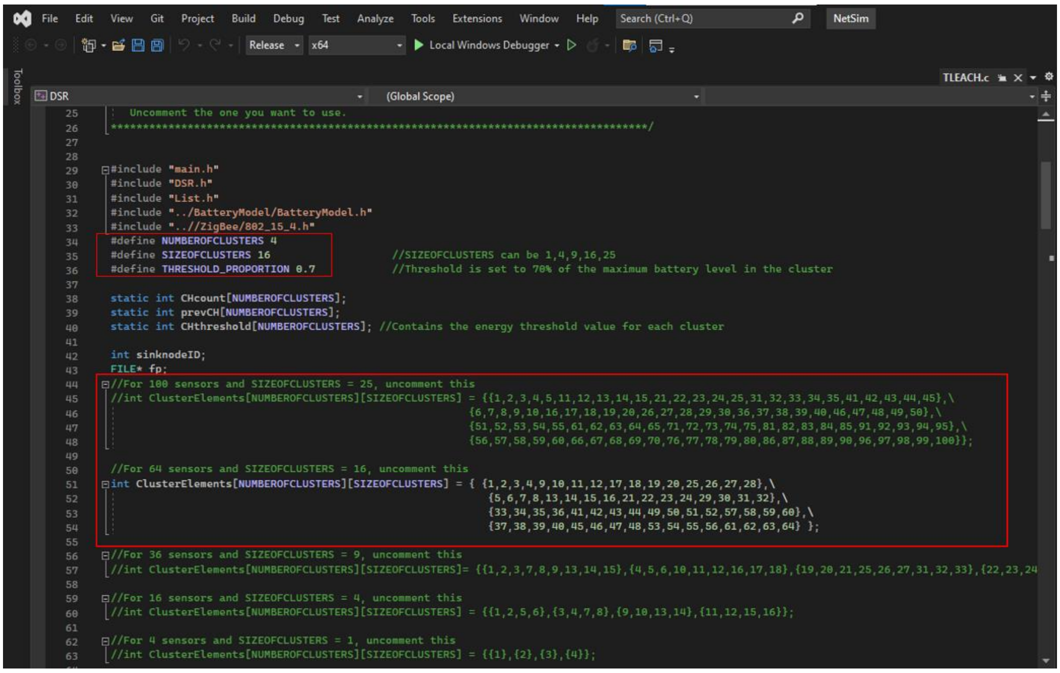


Figure 1: Real World Agriculture Monitoring System.

Implementation of TLEACH in WSN:

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

1. For this implementation of TLEACH, 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 Cluster Element array accordingly.



```
  Uncomment the one you want to use.
********************************************************************************/


#include "main.h"
#include "DSR.h"
#include "List.h"
#include "../BatteryModel/BatteryModel.h"
#include "..//ZigBee/802_15_4.h"
#define NUMBEROFCLUSTERS 4
#define SIZEOFCLUSTERS 16                      //SIZEOFCLUSTERS can be 1,4,9,16,25
#define THRESHOLD_PROPORTION 0.7               //Threshold is set to 70% of the maximum battery level in the cluster

static int CHcount[NUMBEROFCLUSTERS];
static int prevCH[NUMBEROFCLUSTERS];
static int CHthreshold[NUMBEROFCLUSTERS]; //Contains the energy threshold value for each cluster

int sinknodeID;
FILE* fp;
//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}};
```

Figure 2: TLEACH.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.
3. Uncomment the one you want to use. Change the number and size of the clusters in TLeach.c file .Create a new scenario in the same workspace and Drop the Number of sensors in GUI as mentioned in the comment in code.

The file contains the following functions:

- **fn_NetSim_TLEACH_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_TLEACH_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_TLEACH_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_TLEACH_IdentifyCluster();** // This function is used to determine the cluster to which a sensor belongs. It returns the cluster id of the cluster.
- **fn_NetSim_TLEACH_init();** // It is used to initialize TLEACH parameters such as the sink node ID and the initial Threshold value.
- **fn_NetSim_TLEACH_set_threshold(); //** It is used to reduce the threshold value for clusters whenever all sensors in a cluster reach energy levels less than the threshold.

Example:

1. The **TLEACH-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 **TLEACH-in-WSN-Example** from the list of experiments.
2. The example consists of a WSN network with 64 sensors placed uniformly along with a sink node as shown below:

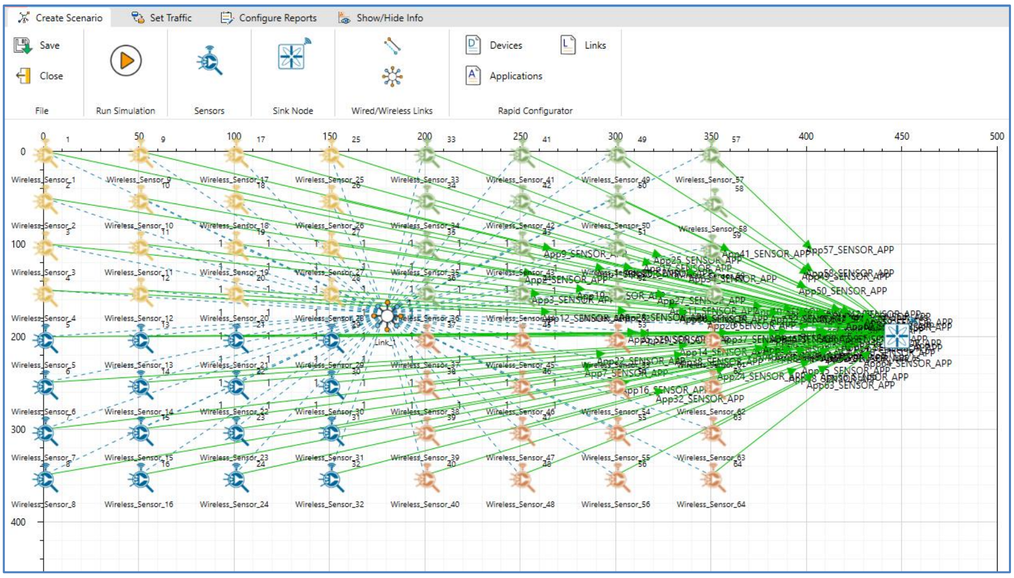


Figure 3: Network Topology

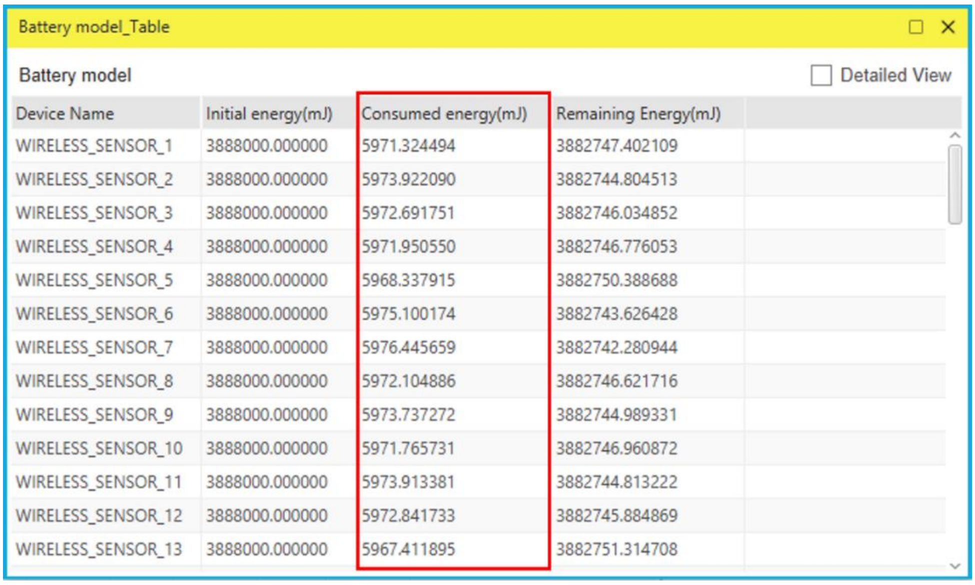
3. Run the Simulation for 500 Seconds.

Results and discussion:

In packet trace. You will note that the sensors directly start transmitting packets without route establishment since the routes are statically defined in TLEACH. You will also note that the cluster heads keep changing dynamically. If the sensor has more remaining energy than the threshold value. It will be elected as cluster head for transmitting the packets to the corresponding clusters.

The battery model table reveals that the consumed energy is significantly lower with TLEACH protocol implementation compared to without TLEACH. This can be observed in the battery model table.

With TLEACH Protocol Implementation:

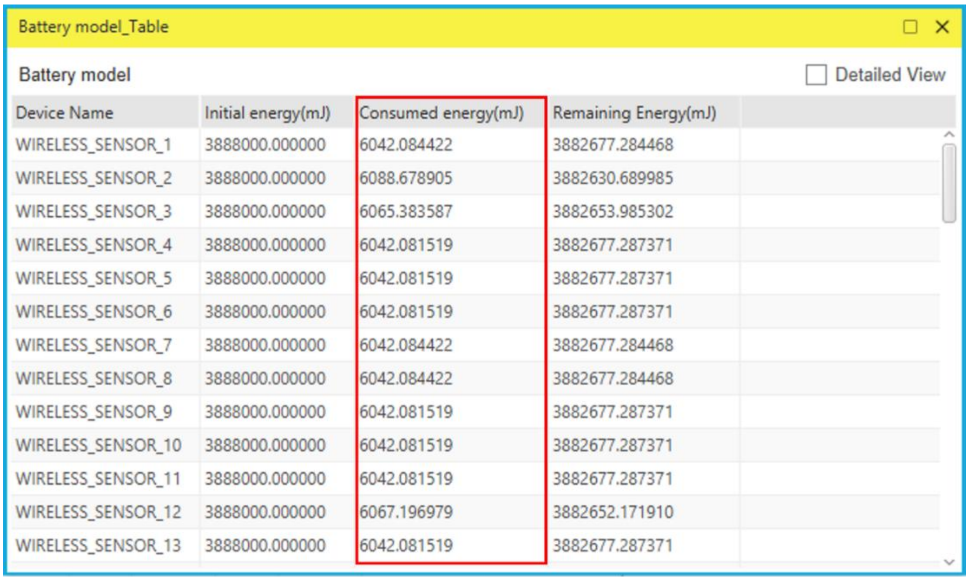


Battery model

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	5971.324494	3882747.402109
WIRELESS_SENSOR_2	3888000.000000	5973.922090	3882744.804513
WIRELESS_SENSOR_3	3888000.000000	5972.691751	3882746.034852
WIRELESS_SENSOR_4	3888000.000000	5971.950550	3882746.776053
WIRELESS_SENSOR_5	3888000.000000	5968.337915	3882750.388688
WIRELESS_SENSOR_6	3888000.000000	5975.100174	3882743.626428
WIRELESS_SENSOR_7	3888000.000000	5976.445659	3882742.280944
WIRELESS_SENSOR_8	3888000.000000	5972.104886	3882746.621716
WIRELESS_SENSOR_9	3888000.000000	5973.737272	3882744.989331
WIRELESS_SENSOR_10	3888000.000000	5971.765731	3882746.960872
WIRELESS_SENSOR_11	3888000.000000	5973.913381	3882744.813222
WIRELESS_SENSOR_12	3888000.000000	5972.841733	3882745.884869
WIRELESS_SENSOR_13	3888000.000000	5967.411895	3882751.314708

Figure 4: Battery model table

Without TLEACH Protocol Implementation:



Battery model

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	6042.084422	3882677.284468
WIRELESS_SENSOR_2	3888000.000000	6088.678905	3882630.689985
WIRELESS_SENSOR_3	3888000.000000	6065.383587	3882653.985302
WIRELESS_SENSOR_4	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_5	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_6	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_7	3888000.000000	6042.084422	3882677.284468
WIRELESS_SENSOR_8	3888000.000000	6042.084422	3882677.284468
WIRELESS_SENSOR_9	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_10	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_11	3888000.000000	6042.081519	3882677.287371
WIRELESS_SENSOR_12	3888000.000000	6067.196979	3882652.171910
WIRELESS_SENSOR_13	3888000.000000	6042.081519	3882677.287371

Figure 5: Battery model table

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