Wireless Energy Harvesting for Internet of Things

Software: NetSim Standard v14.0, Visual Studio 2022

Project Download Link:
https://github.com/NetSim-TETCOS/Wireless-Energy-harvesting-for-IoT-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 to Energy Harvesting:

Among various methods like vibration, light, and thermal energy extraction, wireless energy harvesting (WEH) has proven to be one of the most promising solutions by virtue of its simplicity, ease of implementation, and wide availability. This technology trend is gaining attraction as it offers a fundamental approach to extend the lifespan of batteries. While harvesting from the environmental sources is dependent on the presence of the corresponding energy source, RF (radio frequency) energy harvesting provides unique advantages, being wireless and easily accessible from transmitted energy sources like (TV/radio broadcasters, mobile base stations, and handheld radios), It's cost-effective and allows for compact implementations.

Components of a WEH-Enabled Sensor Device:

A typical WEH-enabled sensor device comprises key components: an antenna, a transceiver, a wireless energy harvesting (WEH) unit, a power management unit (PMU), a sensor/processor unit, and optionally, an onboard battery.

Calculation of Harvested Power (PH):

The available Harvested power (P_H) is determined by the Friis equation. It is directly proportional to Transmitted power (P_T), Path loss (P_L), Transmitter antenna gain (G_T), Receiver antenna gain (G_R), the Power conversion efficiency of the converter (PCE_H), and the square of the wavelength (λ), and is inversely proportional to the square of the communication distance (r) between the source and the device.

Energy Components and Distribution:

The energy consumed by the device can be categorized into communication energy (listening, receiving, and transmitting) and computation energy (processing and sensing).

- Listening energy (E_{LS})
- Receiver energy (E_{RX})
- Transmitter energy (E_{TX})
- Processing energy (E_{PR})
- Sensing energy (E_{SN})

To capture the energy distribution among the aforementioned energy consumers, weighting coefficients, $\alpha_{LS} > \alpha_{TX} > \alpha_{RX} > \alpha_{PR} > \alpha_{SN}$ are assigned to them. The total average energy consumption $E_D = \alpha_{LS}E_{LS} + \alpha_{TX}E_{TX} + \alpha_{RX}E_{RX} + \alpha_{PR}E_{PR} + \alpha_{SN}E_{SN}$ is then calculated as the sum of the product of these coefficients and their respective energy components.

Battery Storage and Harvested Energy:

The total energy stored in the device's battery is denoted as E_B. On the other hand, the available harvested energy per active-duty cycle is represented by E_H.

Topology Considerations:

We assume constant energy consumption for the receiver, processor, and sensor. However, the energy consumption of the transmitter (E_{TX}) is directly proportional to the square of the distance (r^2_{ij}) ,where r_{ij} is the distance between the originating device (j) and the sink node (i), particularly in a ring topology or multihop topology.The harvested energy (E_H) is inversely proportional to r^2_{ij} (here j is the sink node and $r_{ij} = r_{ji}$).

IEEE Ref Paper:

Wireless Energy Harvesting for the Internet of Things
P. Kamalinejad C. Mahapatra; Z. Sheng ; S. Mirabbasi ; V. C. M. Leung ; Y. L. Guan
IEEE COMMUNICATIONS MAGAZINE · JUNE 2015

Real-World Context:

The advent of technology has revolutionized the agricultural landscape, giving rise to Smart Agriculture Monitoring Systems (SAMS) that harness the power of advanced technology to optimize farming practices, enhance crop yields, and promote sustainable agriculture. This discussion focuses on the impact of energy harvesting on wireless sensor nodes within SAMS, utilizing NetSim as a simulation tool.

Without Energy Harvesting

Smart Agriculture Monitoring Systems (SAMS) without energy harvesting rely on traditional power sources, such as batteries, to operate wireless sensor nodes. These batteries have a limited lifespan and require periodic replacement, increasing maintenance costs and environmental concerns due to battery disposal. Additionally, battery depletion can lead to disruptions in data collection, affecting the effectiveness of SAMS in monitoring crop health and environmental conditions

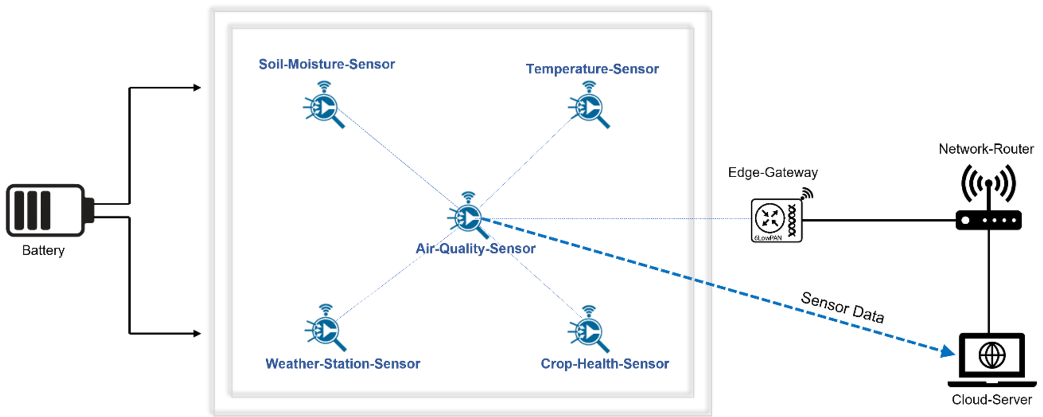


Figure 1: Real world scenario for Smart agriculture system Without Energy harvesting

With Energy Harvesting

The integration of energy harvesting technologies into Smart Agriculture Monitoring Systems (SAMS) revolutionizes the system by introducing sustainable energy sources for wireless sensor nodes. This transformation offers several advantages, including reduced maintenance costs, enhanced environmental sustainability, and uninterrupted data collection. Energy harvesting eliminates the need for frequent battery replacements, significantly lowering maintenance expenses. By minimizing battery usage, Continuous power supply from energy harvesting ensures uninterrupted data collection from wireless sensor nodes, providing real-time insights into crop health and environmental conditions.

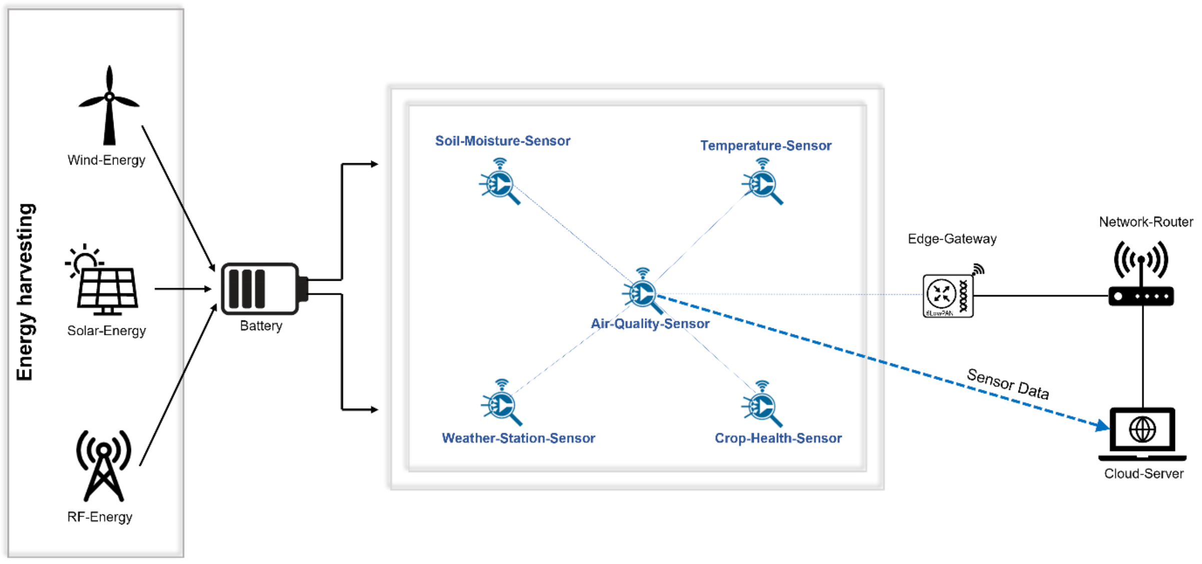


Figure 2: Real world scenario for Smart agriculture system With Energy harvesting

Wireless Energy Harvesting overview

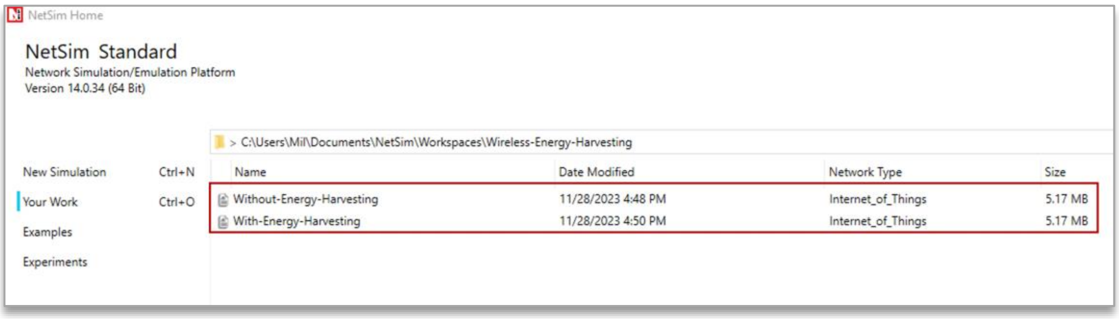
- WEH significantly extends the operational lifespan of IoT devices, reducing the frequency of battery replacements and associated maintenance costs.
- By eliminating the need for frequent battery replacements, WEH minimizes maintenance expenses and enhances the overall efficiency of IoT networks.
- WEH promotes environmental sustainability by reducing battery usage and disposal, mitigating the environmental impact of IoT devices.
- WEH facilitates the deployment of IoT devices in remote or inaccessible areas, expanding the reach of IoT networks and providing valuable insights into diverse environment

The Role of NetSim Simulator

NetSim serves as a valuable tool for assessing the effectiveness of energy harvesting technologies in wireless sensor networks. By providing a comprehensive simulation environment, NetSim enables researchers and engineers to evaluate various system parameters and optimize energy harvesting performance. This capability empowers them to compare the performance of wireless sensor networks with and without energy harvesting, assess the impact of energy harvesting efficiency, optimize energy harvesting placement, analyze the impact of energy harvesting on network stability, and investigate the scalability of energy harvesting. Through these simulations, NetSim plays a critical role in developing optimized and sustainable solutions for wireless sensor networks.

COMPARATIVE ANALYSIS:

1. The **Energy_Harvesting_IOT_Workspace** includes sample network configuration files, namely Without-energy-harvesting and With-energy-harvesting, which are pre-saved.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 14.0.34 (64 Bit)

New Simulation Ctrl+N
Your Work Ctrl+O
Examples
Experiments

> C:\Users\Mil\Documents\NetSim\Workspaces\Wireless-Energy-Harvesting

Name	Date Modified	Network Type	Size
Without-Energy-Harvesting	11/28/2023 4:48 PM	Internet_of_Things	5.17 MB
With-Energy-Harvesting	11/28/2023 4:50 PM	Internet_of_Things	5.17 MB

Figure 3: Sample configuration files

2. We will now open the **“Without-Energy-Harvesting”** sample.
3. The network scenario consists of 16 sensors, a LoWPAN Gateway, a Router and a Wired Node as shown below.

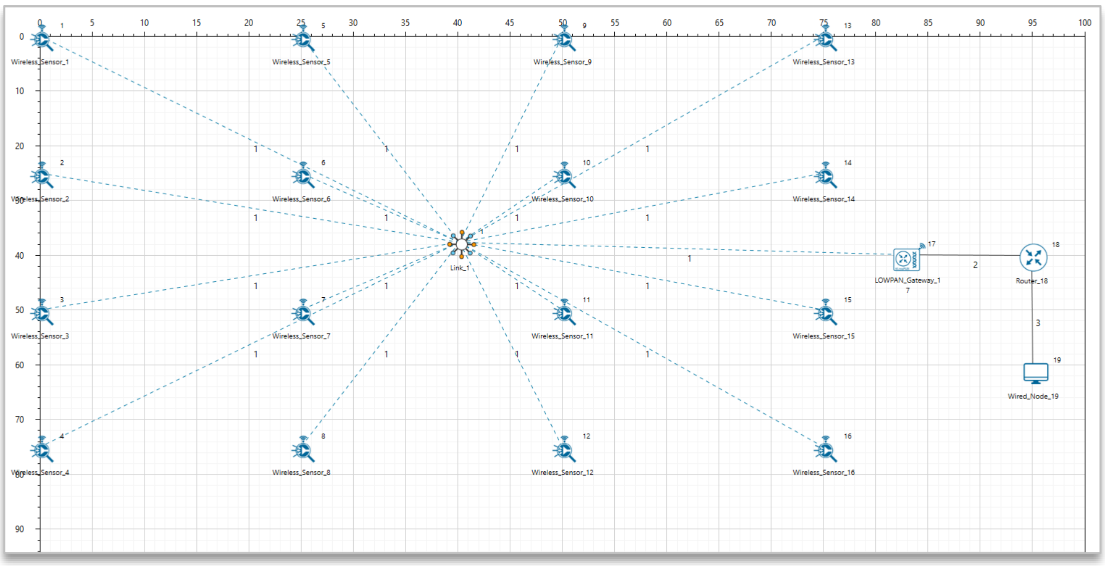
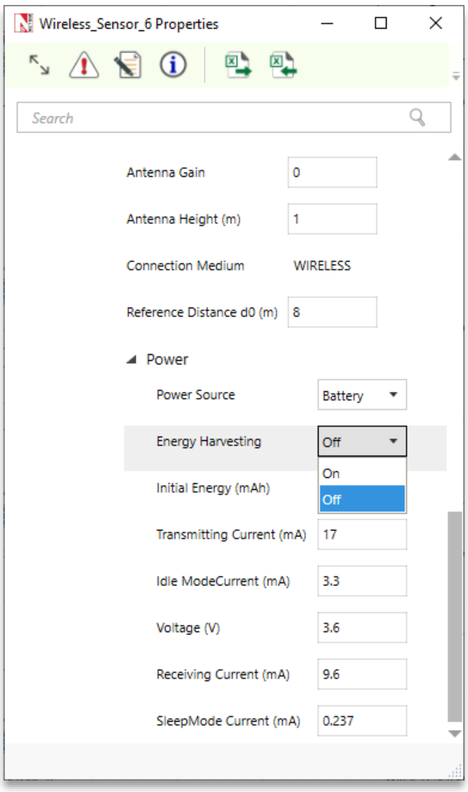


Figure 4: Energy Harvesting Network Topology

4. Here, the Energy Harvesting parameter has been set to OFF in all Sensor nodes. To enable or disable the energy harvesting setting, users can navigate to Interface(ZigBee)->Physical layer -> IEE802.15.4 ->Power ->Set Power source to Battery of the sensor nodes, as shown below



Wireless_Sensor_6 Properties

Search

Antenna Gain	0
Antenna Height (m)	1
Connection Medium	WIRELESS
Reference Distance d0 (m)	8

Power

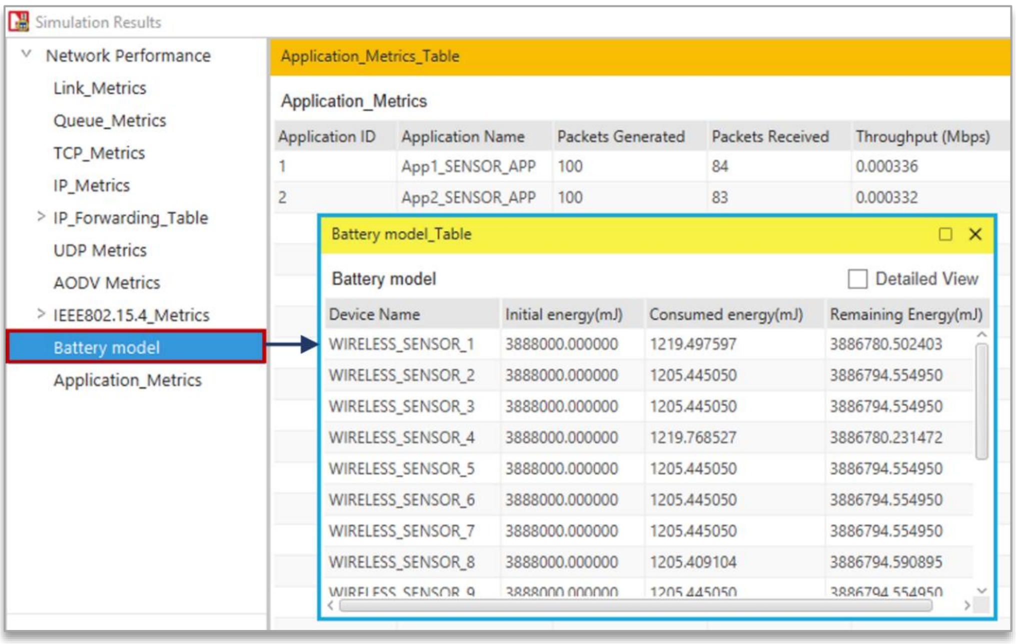
Power Source	Battery
Energy Harvesting	Off (On / Off)
Initial Energy (mAh)	
Transmitting Current (mA)	17
Idle ModeCurrent (mA)	3.3
Voltage (V)	3.6
Receiving Current (mA)	9.6
SleepMode Current (mA)	0.237

Figure 4: Energy Harvesting parameter

5. Run Simulation for 100 seconds and save the simulation results.

Results and Discussion

After the simulation is completed, the NetSim result dashboard will open. Navigate to the Battery model under IEEE802.15.4_Metrics. This will display Battery model table which is part of NetSim Simulation Results window.



Application_Metrics_Table

Application_Metrics

Application ID	Application Name	Packets Generated	Packets Received	Throughput (Mbps)
1	App1_SENSOR_APP	100	84	0.000336
2	App2_SENSOR_APP	100	83	0.000332

Battery model_Table

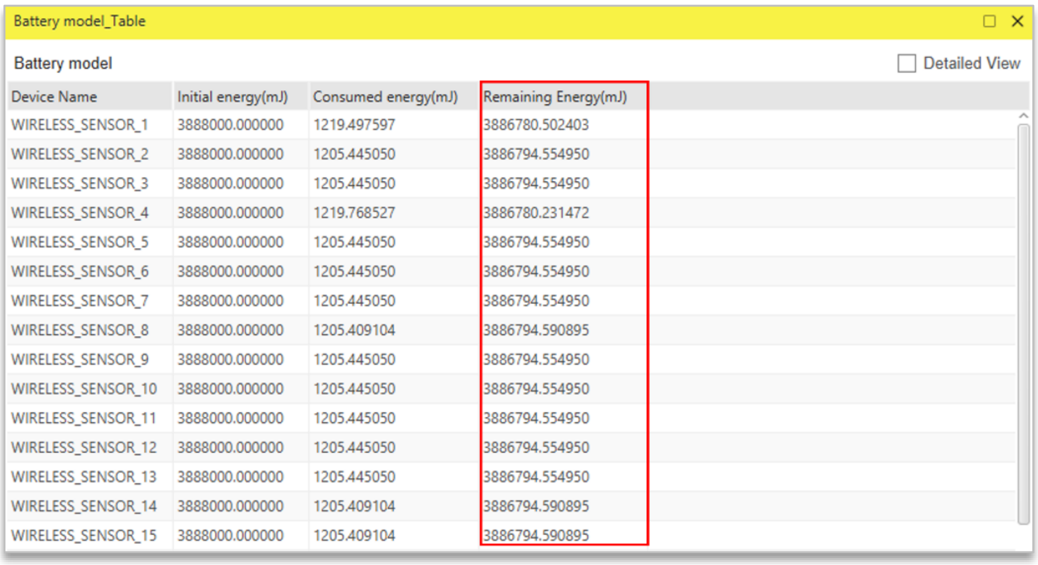
Battery model ☐ Detailed View

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	1219.497597	3886780.502403
WIRELESS_SENSOR_2	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_3	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_4	3888000.000000	1219.768527	3886780.231472
WIRELESS_SENSOR_5	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_6	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_7	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_8	3888000.000000	1205.409104	3886794.590895
WIRELESS_SENSOR_9	3888000.000000	1205.445050	3886794.554950

Figure 5: Battery model table from Netsim result dashboard

WITHOUT ENERGY HARVESTING:

In the Battery model table, you can observe the results for scenario without energy harvesting. This table provides detailed insights into the energy consumption of each sensor node

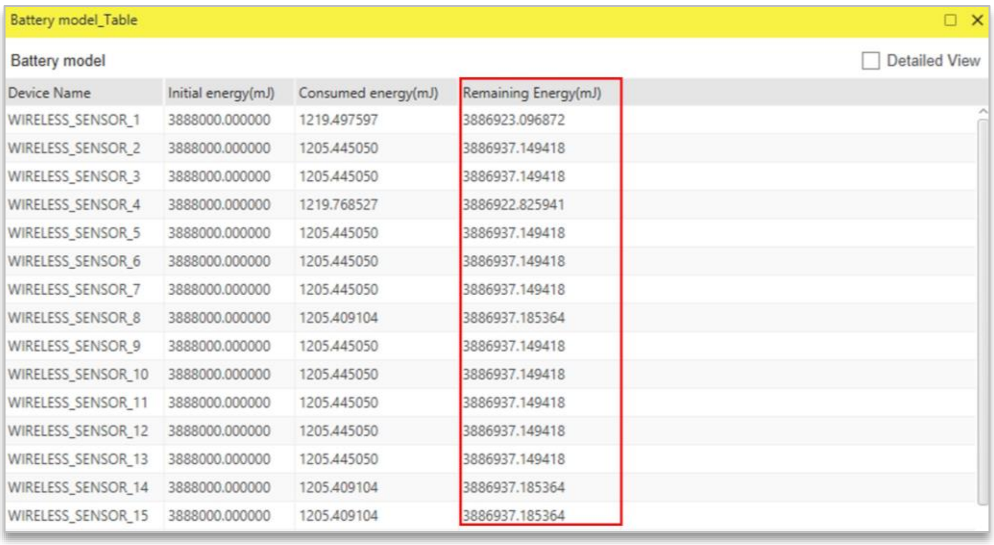


Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	1219.497597	3886780.502403
WIRELESS_SENSOR_2	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_3	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_4	3888000.000000	1219.768527	3886780.231472
WIRELESS_SENSOR_5	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_6	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_7	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_8	3888000.000000	1205.409104	3886794.590895
WIRELESS_SENSOR_9	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_10	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_11	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_12	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_13	3888000.000000	1205.445050	3886794.554950
WIRELESS_SENSOR_14	3888000.000000	1205.409104	3886794.590895
WIRELESS_SENSOR_15	3888000.000000	1205.409104	3886794.590895

Figure 6: Battery model table without energy harvesting

WITH ENERGY HARVESTING:

Now, open and run the 'With-Energy-Harvesting' sample, where Energy Harvesting is enabled for all sensor nodes. Upon comparing the remaining energy levels with the "without-energy-harvesting" Scenario, you will observe increases in the working capability of sensors



Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)
WIRELESS_SENSOR_1	3888000.000000	1219.497597	3886923.096872
WIRELESS_SENSOR_2	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_3	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_4	3888000.000000	1219.768527	3886922.825941
WIRELESS_SENSOR_5	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_6	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_7	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_8	3888000.000000	1205.409104	3886937.185364
WIRELESS_SENSOR_9	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_10	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_11	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_12	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_13	3888000.000000	1205.445050	3886937.149418
WIRELESS_SENSOR_14	3888000.000000	1205.409104	3886937.185364
WIRELESS_SENSOR_15	3888000.000000	1205.409104	3886937.185364

Figure 7 : Battery Model table with energy harvesting

Simulations can be performed for different values of E_H Fraction which may vary as per the efficiency of the Energy Harvesting unit.

Note: You can observe slight variation in the remaining energy with and without energy harvesting as the simulation time is in seconds.

Appendix: NetSim source code modifications

Changes to Battery Model.h within Battery Model project

```
/* We implemented the Batter Model*/

#ifndef _NETSIM_BATTERY_MODEL_H_
#define _NETSIM_BATTERY_MODEL_H_
#ifdef  __cplusplus
extern "C" {
#endif

#ifndef _BATTERY_MODEL_CODE_
#pragma comment(lib,"BatteryModel.lib")
        typedef void* ptrBATTERY;
#endif

        _declspec(dllexport) ptrBATTERY battery_find(NETSIM_ID d,

NETSIM_ID in);
        _declspec(dllexport) void battery_add_new_mode(ptrBATTERY battery, int mode, double current,
char* heading);
        _declspec(dllexport) ptrBATTERY battery_init_new(NETSIM_ID deviceId, NETSIM_ID interfaceId,
double initialEnergy, double voltage, double dRechargingCurrent);

        _declspec(dllexport) bool battery_set_mode(ptrBATTERY battery, int mode, double time);
        _declspec(dllexport) void battery_animation();
        _declspec(dllexport) void battery_metrics(PMETRICSWRITER metricsWriter);
        _declspec(dllexport) double battery_get_remaining_energy(ptrBATTERY battery);
        _declspec(dllexport) int battery_energy_harvesting(ptrBATTERY battery, double eh_energy);
        _declspec(dllexport) double battery_get_consumed_energy(ptrBATTERY battery, int mode);

#ifdef  __cplusplus
}
#endif
#endif //_NETSIM_BATTERY_MODEL_H_
```

Changes to double battery_get_remaining_energy (), Battery Model.c within Battery Model project

```
_declspec(dllexport) double battery_get_remaining_energy(ptrBATTERY battery)
{
        return battery->remainingEnergy;
}
_declspec(dllexport) int battery_energy_harvesting(ptrBATTERY battery, double eh_energy)
{
        double eh_energy_mJ = eh_energy * ((pstruEventDetails->dEventTime - battery-
>modeChangedTime) / 1000000);
        battery->remainingEnergy += eh_energy_mJ;
}
```

Changes code to ChangeRadioState.c, within Zigbee project at the end of the file

```
#define EH_FRACTION 0.1
// EH_FRACTION is the fraction of the received signal energy that can be
// captured and harvested by the sensor.
```

```
int calculate_eh(NETSIM_ID dev1, NETSIM_ID dev2)
{
        double rx_pwr = GET_RX_POWER_mw(dev1, dev2, pstruEventDetails->dEventTime);
        double eh_energy = EH_FRACTION * rx_pwr;
        ptrBATTERY battery = WSN_PHY(dev2)->battery;
        if (battery)
                battery_energy_harvesting(battery, eh_energy);
}
```

Changes code to int fn_NetSim_Zigbee_Run(), 802_15_4.c file**, within Zigbee project**

```
case UPDATE_MEDIUM:
{
double dtime=pstruEventDetails->dEventTime;
NETSIM_ID nLink_Id, nConnectionID, nConnectionPortID, nLoop;
NETSIM_ID nTransmitterID;

nTransmitterID = pstruEventDetails->nDeviceId;

ZIGBEE_CHANGERADIOSTATE(nTransmitterID, WSN_PHY(nTransmitterID)->nRadioState, RX_ON_IDLE);
if(WSN_PHY(nTransmitterID)->nRadioState != RX_OFF)
WSN_MAC(nTransmitterID)->nNodeStatus = IDLE;
nLink_Id = fn_NetSim_Stack_GetConnectedDevice(pstruEventDetails->nDeviceId,pstruEventDetails-
>nInterfaceId,&nConnectionID,&nConnectionPortID);

for(nLoop=1; nLoop<=NETWORK->ppstruNetSimLinks[nLink_Id-1]-
>puniDevList.pstruMP2MP.nConnectedDeviceCount; nLoop++)
{
NETSIM_ID ncon = NETWORK->ppstruNetSimLinks[nLink_Id-1]->puniDevList.pstruMP2MP.anDevIds[nLoop-
1];
if(ncon != pstruEventDetails->nDeviceId)
{
calculate_eh(nTransmitterID, nLoop);
WSN_PHY(ncon)->dTotalReceivedPower -= GET_RX_POWER_mw(nTransmitterID,ncon,pstruEventDetails-
>dEventTime);

if(WSN_PHY(ncon)->dTotalReceivedPower < WSN_PHY(ncon)->dReceiverSensivity)
WSN_PHY(ncon)->dTotalReceivedPower = 0;
}
}
```

This completes the code modifications for energy harvesting.