**Automatic plotting of DTDMA parameters in NetSim**

**Software:** NetSim Standard/Pro v13.0 (32-bit/ 64-bit), Visual Studio 2017/2019.

**Project Download Link:**

<https://github.com/NetSim-TETCOS/NetSim_DTDMA_Plots_v13_0/archive/refs/heads/main.zip>

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

<https://support.tetcos.com/en/support/solutions/articles/14000128666-downloading-and-setting-up-netsim-file-exchange-projects>

**Features**

Using this workspace:

1. Users can plot Pathloss, Shadow Loss, Fading Loss, Total Loss, Rx\_Power, SNR, and BER vs. time using NetSim Plot.
2. Users can log Pathloss, Shadow Loss, Fading Loss, Total Loss, Rx\_Power, SNR, and BER vs. time using NetSim Plot., with time stamps, to a CSV log file.
3. Users need to provide a file-based input (per a certain format) at the start of simulation for the parameters to be plotted or logged.
4. The plots are unique to Each Wireless Nodes pair.
5. The log entries are unique to Each Wireless Nodes pair.
6. Parameters are logged every slot time (1ms) and plotted.
7. There is no restriction in NetSim on the number of Wireless Nodes in the network.

**Example**

In the below scenario

* Two Wireless Nodes are dropped on the grid and both Fast Fading and ShadowFadingLoss are enabled.
* Wireless Node-2 moves in a straight line away from the Wireless Node\_1.
* The network is simulated for 50 s.

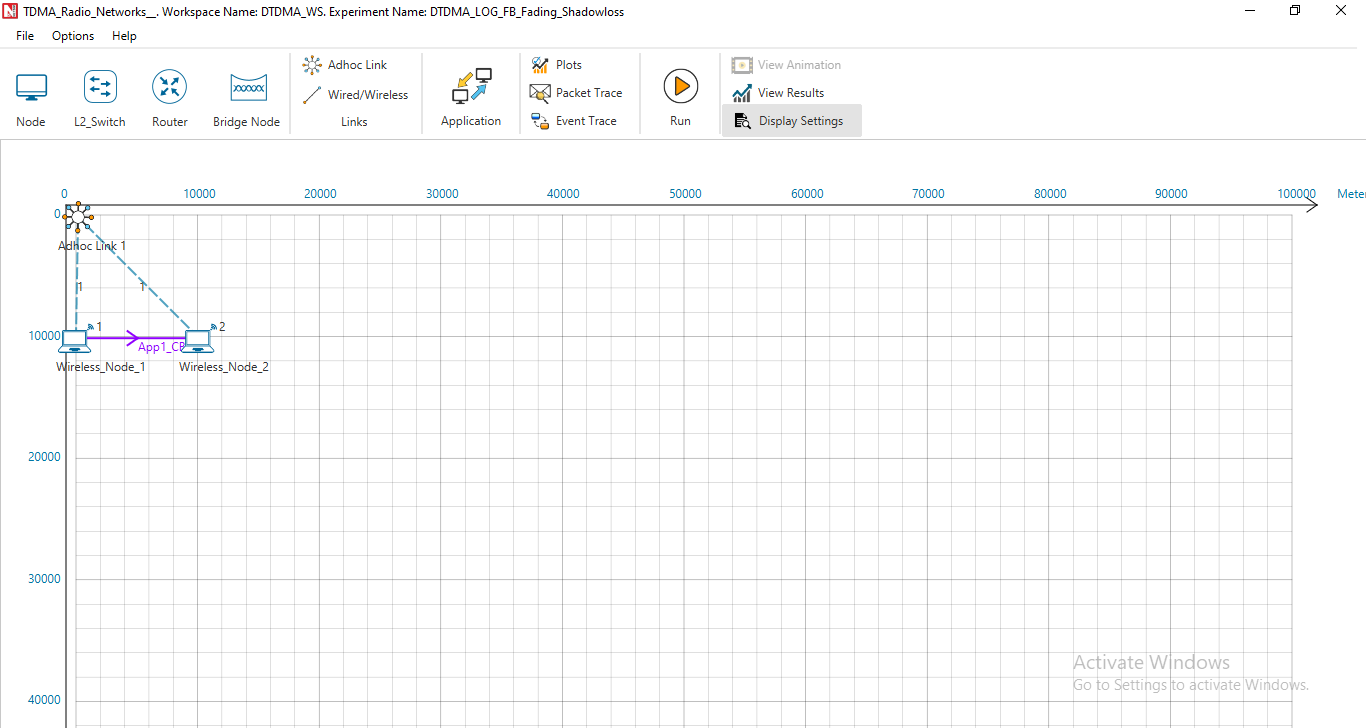


Figure 1: Network Topology in this experiment

* Upon running the simulation, a text file will open for the user to input the parameters and devices (tx-rx pair) for which parameters need to be logged and plotted. The input is per the format of <parameter>, <device1>, <device2> in the text file. To log Wireless Node data flow would be between two Wireless Nodes. In case of multiple Nodes, this input can be given for various Wireless node pairs. Inputs are not case sensitive.
* For the above example, the input text file is as follows.

TOTALLOSS, Wireless\_Node\_1, Wireless\_Node\_2

PATHLOSS, Wireless\_Node\_1, Wireless\_Node\_2

SHADOWLOSS, Wireless\_Node\_1, Wireless\_Node\_2

FADINGLOSS, Wireless\_Node\_1, Wireless\_Node\_2

SNR, Wireless\_Node\_1, Wireless\_Node\_2

RX\_POWER, Wireless\_Node\_1, Wireless\_Node\_2

BER, Wireless\_Node\_1, Wireless\_Node\_2

* Once the simulation starts, in the command prompt window it will show a message as “**Please update, Save and close the file and press any key to continue**”.
* Add the parameters to be logged, close the input text file and press any key.
* Simulation starts running.

**Results and discussion**

Upon completion of simulation in the result window users can view the various plots.

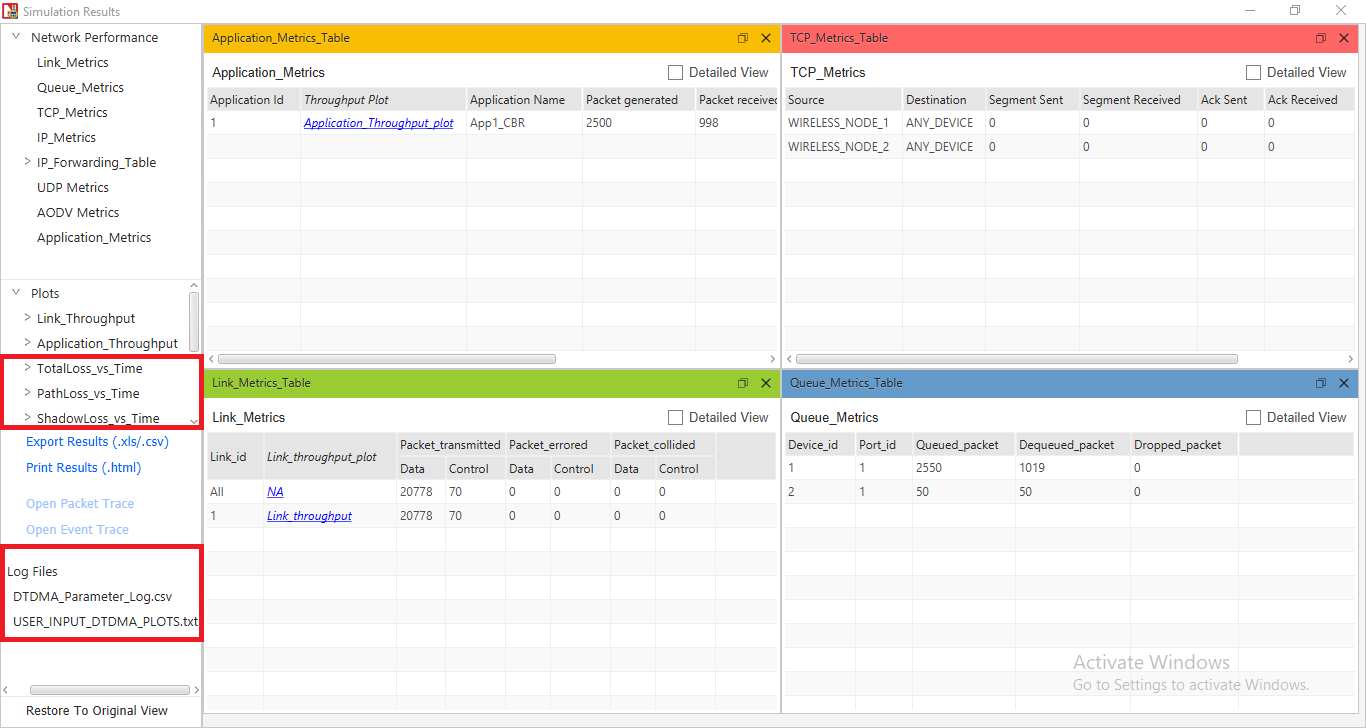


Figure 2: NetSim results dashboard with throughput highlighted

The Pathloss, shadow fading loss, and total loss remains same across the layers

**Result Plots**

1. **Pathloss Plot**

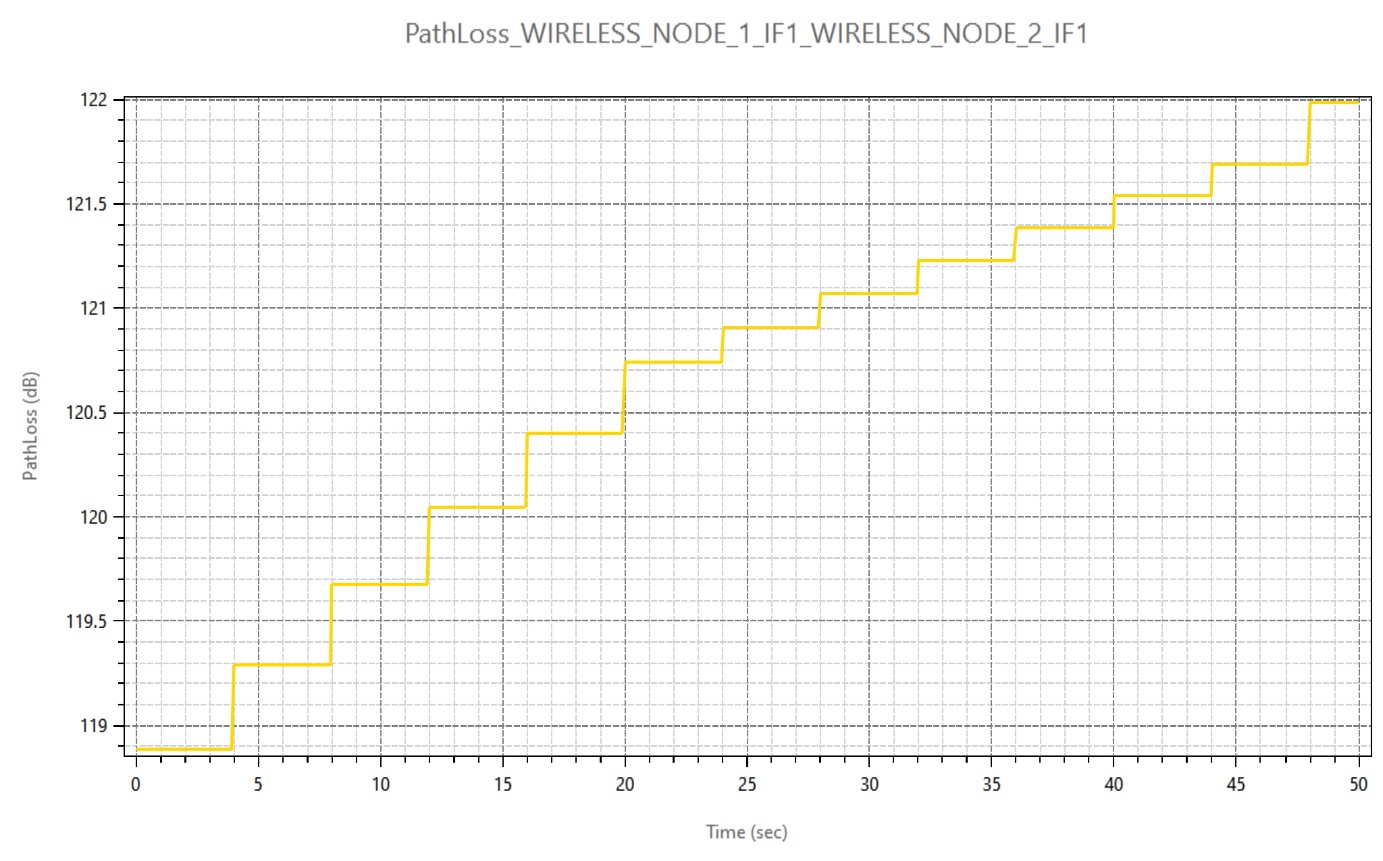


Figure 3: Pathloss Plot in NetSim

1. **Total Loss (Shadow Fading loss plus Path loss)**

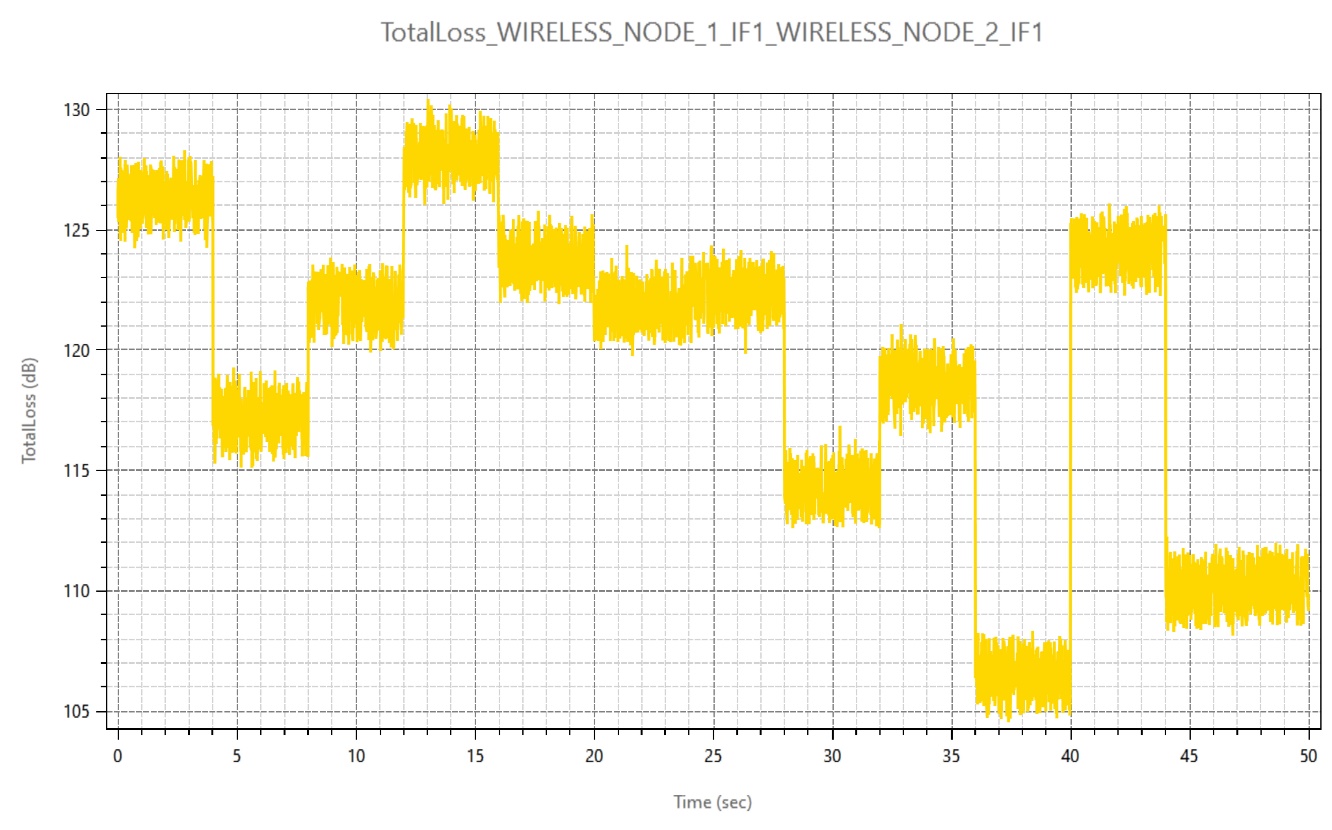


Figure 4: Total Loss (Shadow Fading loss plus Path loss) in NetSim

1. **Shadow Loss**

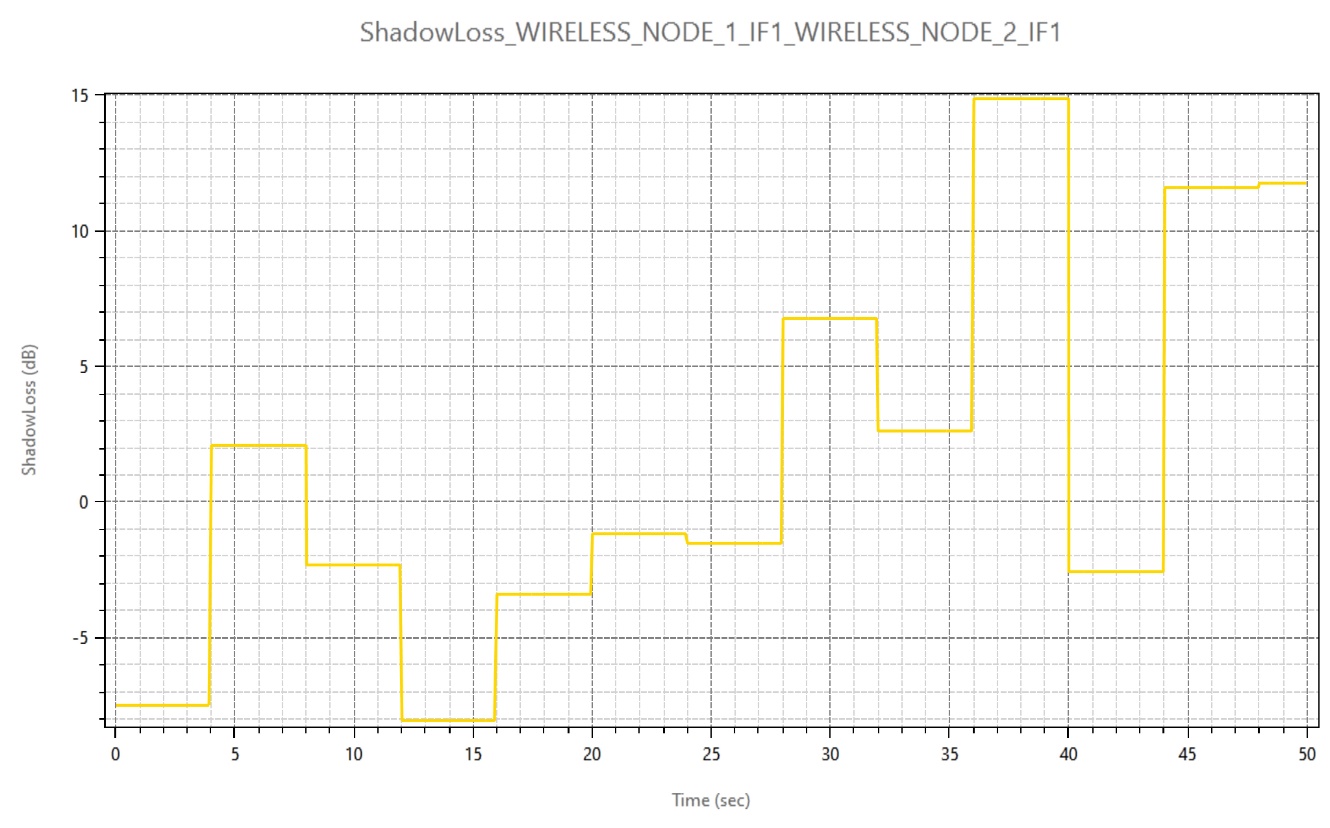
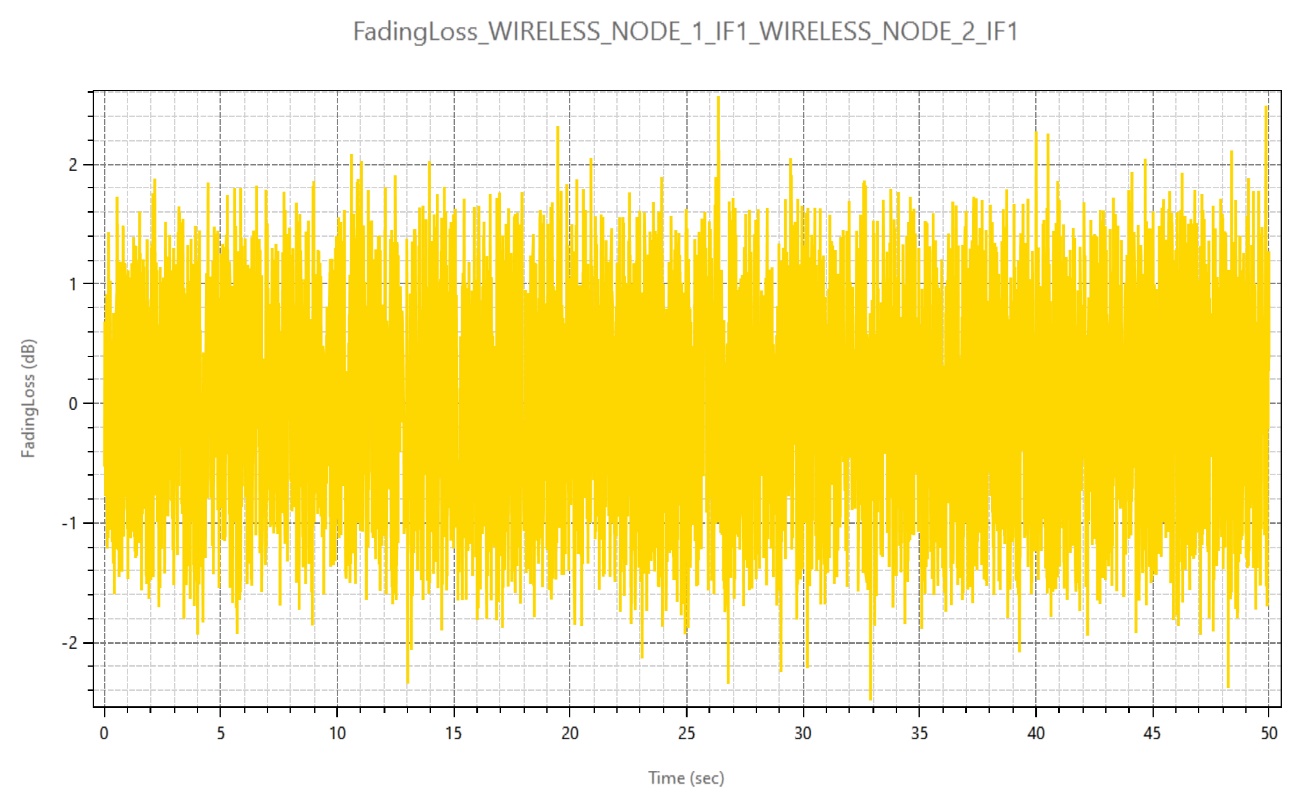


Figure 5: Shadow Fading Loss in NetSim

4. **Fading Loss.**



**Figure 6**: Fading Loss

The plot title is FadingLoss\_WIRELESS\_NODE\_1\_IF1\_WIRELESS\_NODE\_2\_IF1. And the naming convention is <ParameterType>\_WIRELESS\_NODE\_<ID>\_IF<InterfaceID>\_WIRELESS\_NODE\_<ID>\_IF<InterfaceID>

1. **Rx\_Power Plot**

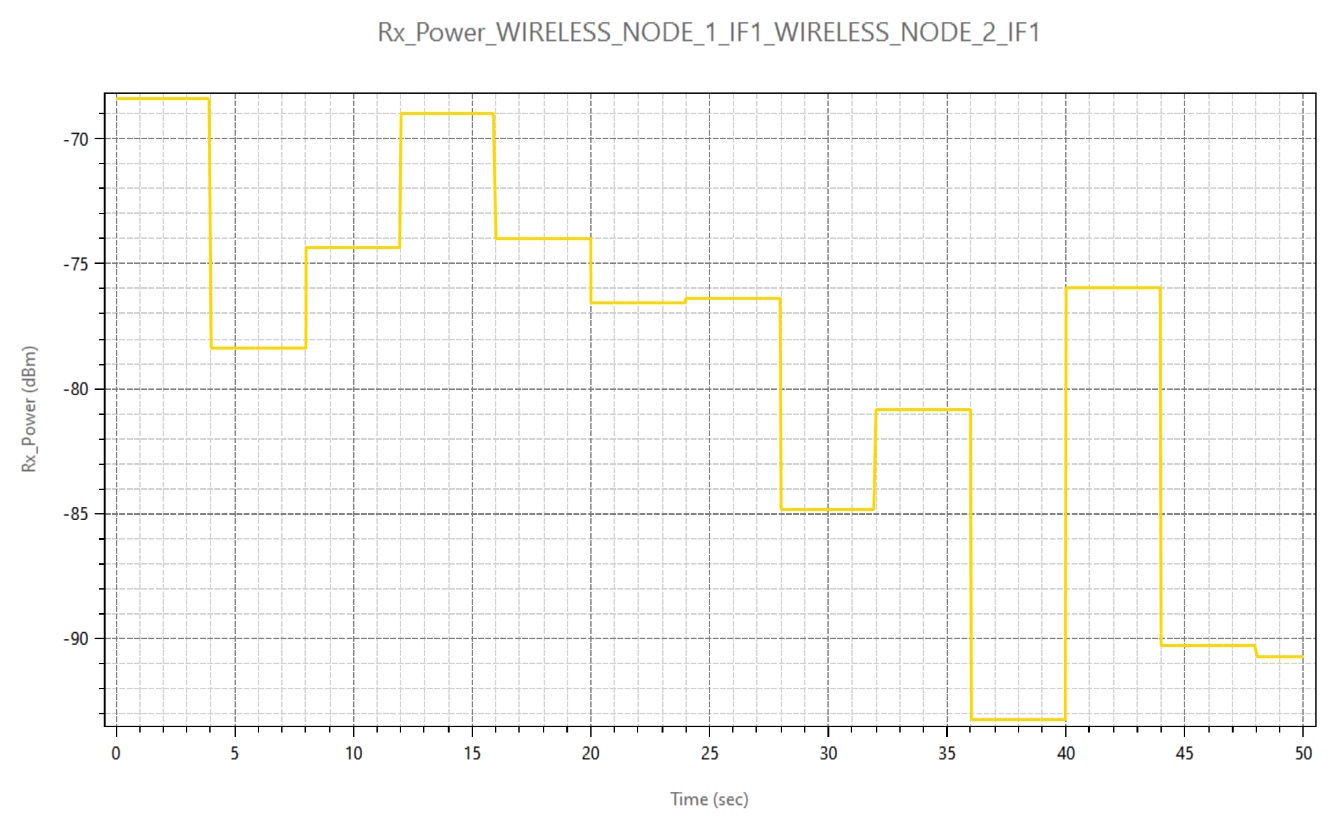
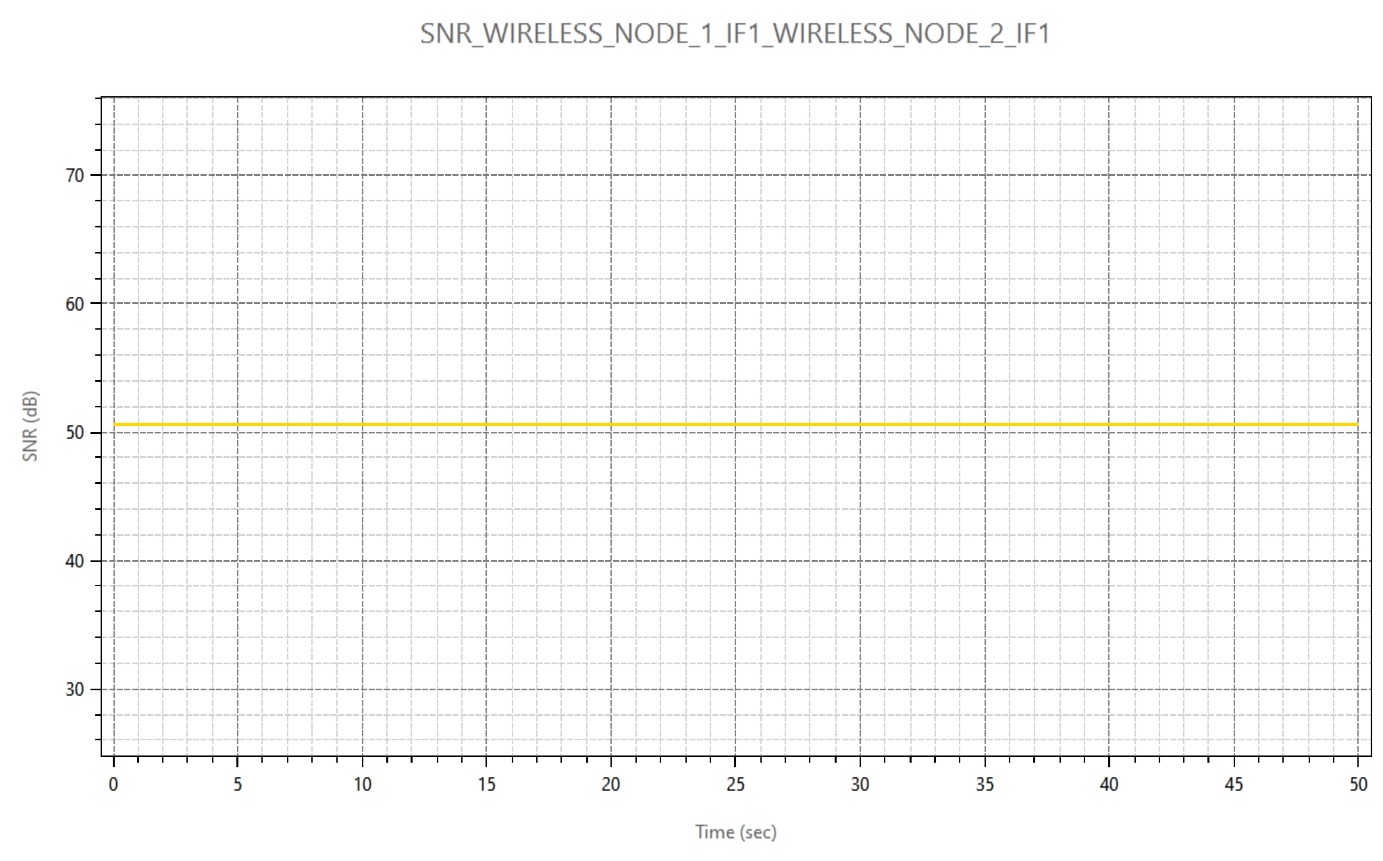
****

Figure 6: Rx\_Power Plot in NetSim.

1. **SNR Plot**



**Figure 7:** SNR Plot in NetSim

**Parameter log file:**

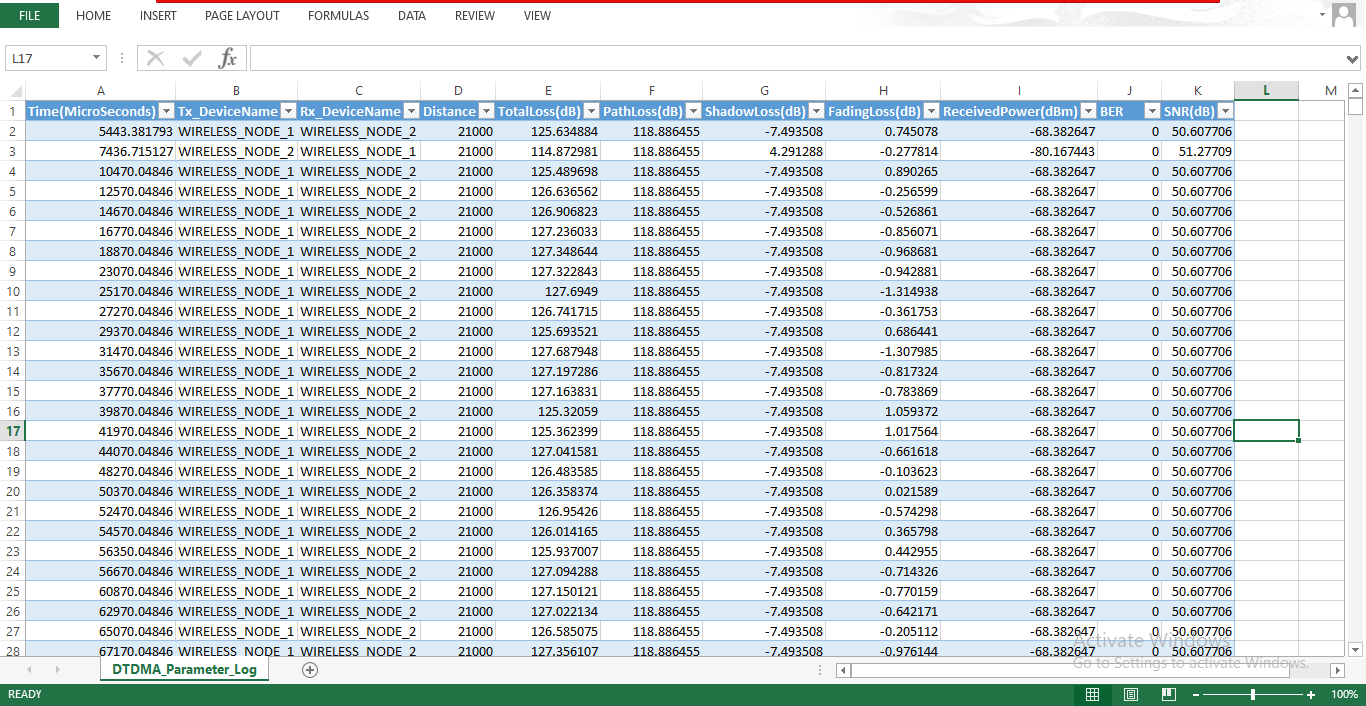


Figure 11: DTDMA Log file parameter

The DTDMA\_Parameter\_Log.csv file logs the details of parameters specified in the input file with respect to time.

**Appendix: NetSim source code modifications**

Open the Source codes in Visual Studio by going to your work-> Workspace Options and Clicking on Open code button.

To the in DTDMA project, DTDMA.c file contain the definitions of the functions that responsible for plotting and logging parameters associated with DTDMA network in NetSim.

The function fn\_NetSim\_DTDMA\_Init\_Plots and fn\_NetSim\_DTDMA\_init\_Parameter\_Log has been called in DTDMA.c file for initializing the plot.

int fn\_NetSim\_DTDMA\_Mobility(NETSIM\_ID nNodeId);

double codingrate\_to\_double\_from\_string(char\* s);

static bool isplotinit = false;

/\*\*

DTDMA Init function initializes the DTDMA parameters.

\*/

\_declspec (dllexport) int fn\_NetSim\_DTDMA\_Init(struct stru\_NetSim\_Network \*NETWORK\_Formal,

NetSim\_EVENTDETAILS \*pstruEventDetails\_Formal,

char \*pszAppPath\_Formal,

char \*pszWritePath\_Formal,

int nVersion\_Type,

void \*\*fnPointer)

{

if (!isplotinit)

{

fn\_NetSim\_DTDMA\_Init\_Plots ();

fn\_NetSim\_DTDMA\_init\_Parameter\_Log ();

isplotinit = true;

}

fn\_NetSim\_DTDMA\_NodeInit();

fn\_NetSim\_DTDMA\_CalulateReceivedPower ();

init\_dtdma\_session();

init\_slot\_formation();

fn\_NetSim\_DTDMA\_InitFrequencyHopping ();

fnMobilityRegisterCallBackFunction (fn\_NetSim\_DTDMA\_Mobility);

fnNodeJoinRegisterCallBackFunction (fnDTDMANodeJoinCallBack);

return 0;

}

The initialization of functions and functions to update the logs for plotting and logging to CSV file has been made as follows in fn\_NetSim\_DTDMA\_Run() function and fn\_NetSim\_DTDMA\_CalulateReceivedPower() function DTDMA\_Phy.c file

case PHYSICAL\_OUT\_EVENT:

{

fn\_NetSim\_DTDMA\_PhysicalOut();

}

break;

case PHYSICAL\_IN\_EVENT:

pstruEventDetails->pPacket->pstruPhyData->nPacketErrorFlag =

fn\_NetSim\_DTDMA\_CalculatePacketError(d, in, pstruEventDetails->pPacket);

//Function calls

double dThemalNoise= 0; //in dBm

double dBandwidth;

double dRx\_Power = GET\_RX\_POWER\_dbm(pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId,d, in);

DTDMA\_NODE\_PHY\* phy = DTDMA\_PHY(pstruEventDetails->pPacket->nTransmitterId, in);

PPROPAGATION\_INFO\*\*\*\* info = propagationHandle;

PPROPAGATION\_INFO pinfo = info [pstruEventDetails->pPacket->nTransmitterId][in][pstruEventDetails->nDeviceId][pstruEventDetails->nInterfaceId];

double fading = propagation\_calculate\_fadingloss (propagationHandle,pstruEventDetails->pPacket->nTransmitterId,in,pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId);

double ber= calculate\_BER(phy->modulation,GET\_RX\_POWER\_dbm(pstruEventDetails->pPacket->nTransmitterId, pstruEventDetails->nInterfaceId,d, in),

phy->dBandwidth);

double snr= dRx\_Power - dThemalNoise;

fn\_NetSim\_DTDMA\_Log\_Parameters(pinfo, fading, snr, ber);

fn\_NetSim\_DTDMA\_add\_PropagationInfo\_Plot\_data(pinfo, fading, snr, ber);

fn\_NetSim\_DTDMA\_add\_Power\_Plot\_data(pinfo, snr, ber);

pstruEventDetails->pPacket->nPacketStatus = pstruEventDetails->pPacket->pstruPhyData->nPacketErrorFlag;

fn\_NetSim\_Metrics\_Add(pstruEventDetails->pPacket);

fn\_NetSim\_WritePacketTrace(pstruEventDetails->pPacket);

if(pstruEventDetails->pPacket->nPacketStatus == PacketStatus\_NoError)

{

pstruEventDetails->nEventType=MAC\_IN\_EVENT;

fnpAddEvent(pstruEventDetails);

}

else

{

fn\_NetSim\_Packet\_FreePacket(pstruEventDetails->pPacket);

}

break;

**DTDMA\_Phy.c**

NETSIM\_ID t, ti, r, ri;

propagationHandle = propagation\_init(MAC\_PROTOCOL\_DTDMA, NULL,DTDMA\_gettxinfo, check\_interference);

parameter\_plot\_info = (ptrplotINFO\*\*\*\*)calloc(NETWORK->nDeviceCount + 1, sizeof \* parameter\_plot\_info);

for (t = 0; t < NETWORK->nDeviceCount; t++)

{

parameter\_plot\_info[t + 1] = (ptrplotINFO\*\*\*)calloc(DEVICE(t + 1)->nNumOfInterface+1, sizeof \* parameter\_plot\_info[t + 1]);

for (ti = 0; ti < DEVICE(t + 1)->nNumOfInterface; ti++)

{

if (!isDTDMAConfigured(t + 1, ti + 1))

continue;

parameter\_plot\_info[t + 1][ti + 1] = (ptrplotINFO\*\*)calloc(NETWORK->nDeviceCount+1, sizeof \* parameter\_plot\_info[t + 1][ti + 1]);

for (r = 0; r < NETWORK->nDeviceCount; r++)

{

parameter\_plot\_info[t + 1][ti + 1][r + 1] = (ptrplotINFO\*)calloc(DEVICE(r + 1)->nNumOfInterface+1, sizeof \* parameter\_plot\_info[t + 1][ti + 1][r + 1]);

for (ri = 0; ri < DEVICE(r + 1)->nNumOfInterface; ri++)

{

if (!isDTDMAConfigured(r + 1, ri + 1))

continue;

parameter\_plot\_info[t + 1][ti + 1][r + 1][ri + 1] = (ptrplotINFO)calloc(1, sizeof \* parameter\_plot\_info[t + 1][ti + 1][r + 1][ri + 1]);

dtdma\_CalculateReceivedPower(t + 1, ti + 1, r + 1, ri + 1);

PPROPAGATION\_INFO\*\*\*\* info = propagationHandle;

PPROPAGATION\_INFO pinfo = info[t + 1][ti + 1][r + 1][ri + 1];

ptrINFO infolog = parameter\_log\_info;

ptrplotINFO\*\*\*\* infoplot = parameter\_plot\_info;

ptrplotINFO info1 = infoplot[pinfo->nTxId][pinfo->nTxInterface][pinfo->nRxId][pinfo->nRxInterface];

// infolog->isParameterlog = false;

info1->isFadingLossPlotEnable = false;

info1->isPathLossPlotEnable = false;

info1->isShadowLossPlotEnable = false;

info1->isTotalLossPlotEnable = false;

info1->isrxPowerPlotEnable = false;

fn\_NetSim\_DTDMA\_init\_PropagationInfo\_Plots(pinfo);

fn\_NetSim\_DTDMA\_init\_Power\_Plots(pinfo);

}

}

}

}

return 0;

}