LTE Discontinuous Reception (DRX)

Software Recommended: NetSim Standard v12.0 (32/64 bit), Visual Studio 2019

Follow the instructions specified in the following link to clone/download the project folder from GitHub using Visual Studio:
https://tetcos.freshdesk.com/support/solutions/articles/14000099351-how-to-clone-netsim-file-exchange-project-repositories-from-github-

Other tools such as GitHub Desktop, SVN Client, Sourcetree, Git from the command line, or any client you like to clone the Git repository.

Note: It is recommended not to download the project as an archive (compressed zip) to avoid incompatibility while importing workspaces into NetSim.

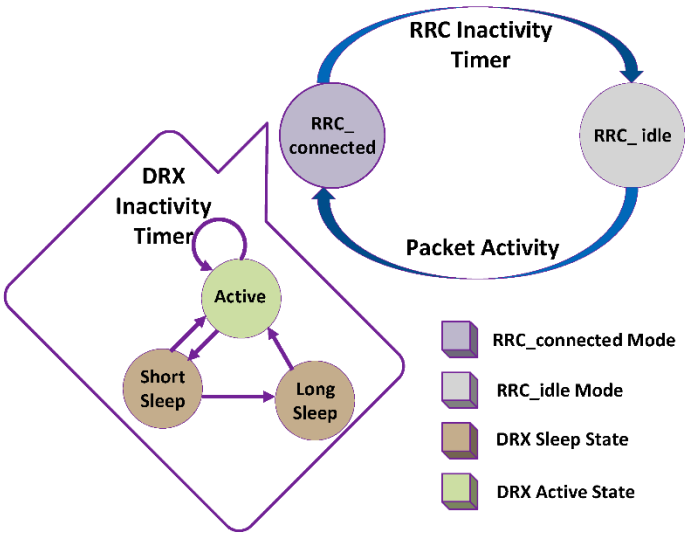
Secure URL for the GitHub repository:

https://github.com/NetSim-TETCOS/LTE_DRX_v12.0.git

Introduction:

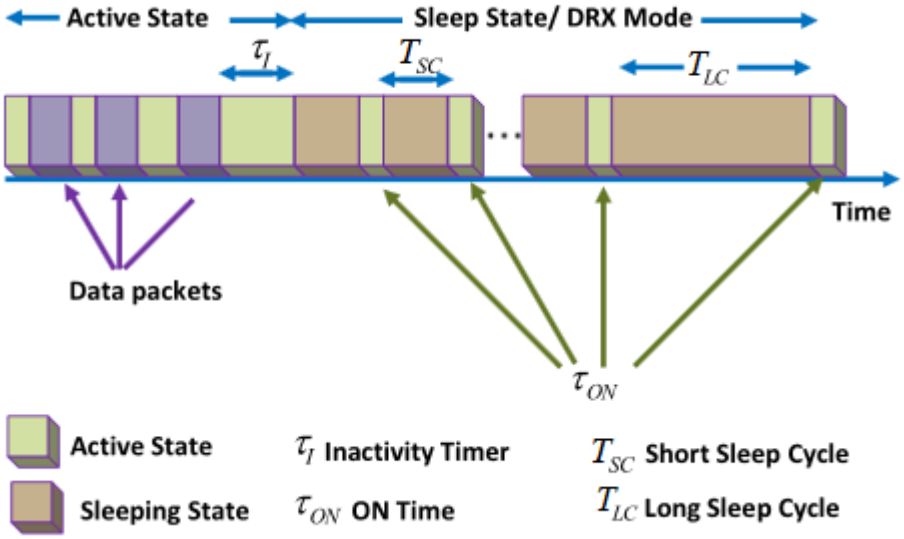
Reference:
https://www.sharetechnote.com/html/Handbook_LTE_DRX.html
https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3579

In normal operation, UE has to be awake all the time and monitor PDCCH for every subframe meaning that it has to be awake all the time since it doesn't know exactly when the network will transmit the data for it. Logically there is no problem with this, but there would be a practical problem. It is power consumption issue on UE side. If UE is always up even when there is no data being transmitted to it from the network, it would be wasting the energy. Then what would be the solution to save the energy on UE side. There may be several ways, but one of the most common way is to use DRX. DRX is a mechanism in which UE gets into sleep mode for a certain period and wake up for another period of time.



We design DRX as a three-state model as shown above. These states are (1) active state, (2) ON state, and (3) sleep state. The UE can receive packets during the active state. In case of no packet in the buffer, UE waits until the expiry of inactivity timer. If UE receives an intimation of packets arrival before

the expiry of inactivity timer, then inactivity timer gets restarted. In case if no packet arrives and inactivity timer completes its countdown, UE switches to sleep state. After the completion of sleep time, the UE switches to ON state to monitor the PDCCH for new packets in the buffer. If the buffer has any packet to be served, the UE transits from ON state to active state; otherwise, the UE continues to sleep and saves the power.



Shown above is a timing diagram of DRX algorithm in RRC_Con mode. The DRX algorithm starts with inactivity timer (TI). The TI continues to countdown from the time instant when all packets in the buffer are served to UE. The TI restarts when a new packet arrives. If no new packet arrives before the expiry of TI, the UE enters into short sleep cycle(TSC). The moment when Tsc expires, the UE switches to listening mode (T ON) to monitor PDCCH. If there is any packet indication during T ON, the UE shifts to active mode, else UE continues to sleep. At the expiry of short sleep timer (TN SC),UE switches to long sleep state (TLC). The UE will remain in this state until TLC gets expired. Similar to the short sleep state, listening state (T ON) is activated after the expiry of long sleep cycle to monitor the PDCCH. In case of any new packet indication,TLC is terminated and UE transits to the active state. This process is repeated in RRC_Con mode. Table 1 shows the list of parameters in RRC_Con, which can be controlled and configured through RRC. The optimum configurations of DRX parameters in RRC_Con mode might improve the Quality of Service (QoS).

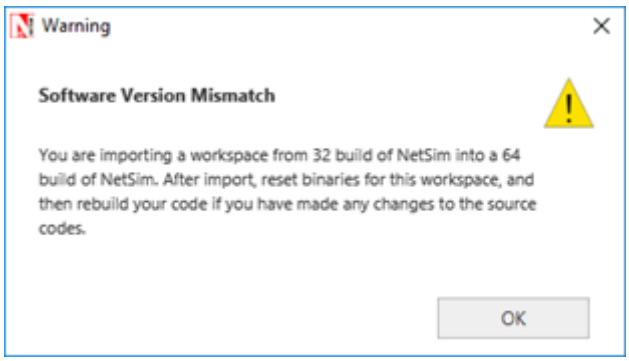
The source codes of LTE project, which is part of NetSim protocol source codes, is modified for this implementation. The LTE DRX project workspace which contains the modified code and related network scenarios which were considered for the implementation can be accessed as explained below.

Follow the instructions specified in the following link to clone/download the project folder from GitHub using Visual Studio:
https://tetcos.freshdesk.com/support/solutions/articles/14000099351-how-to-clone-netsim-file-exchange-project-repositories-from-github-

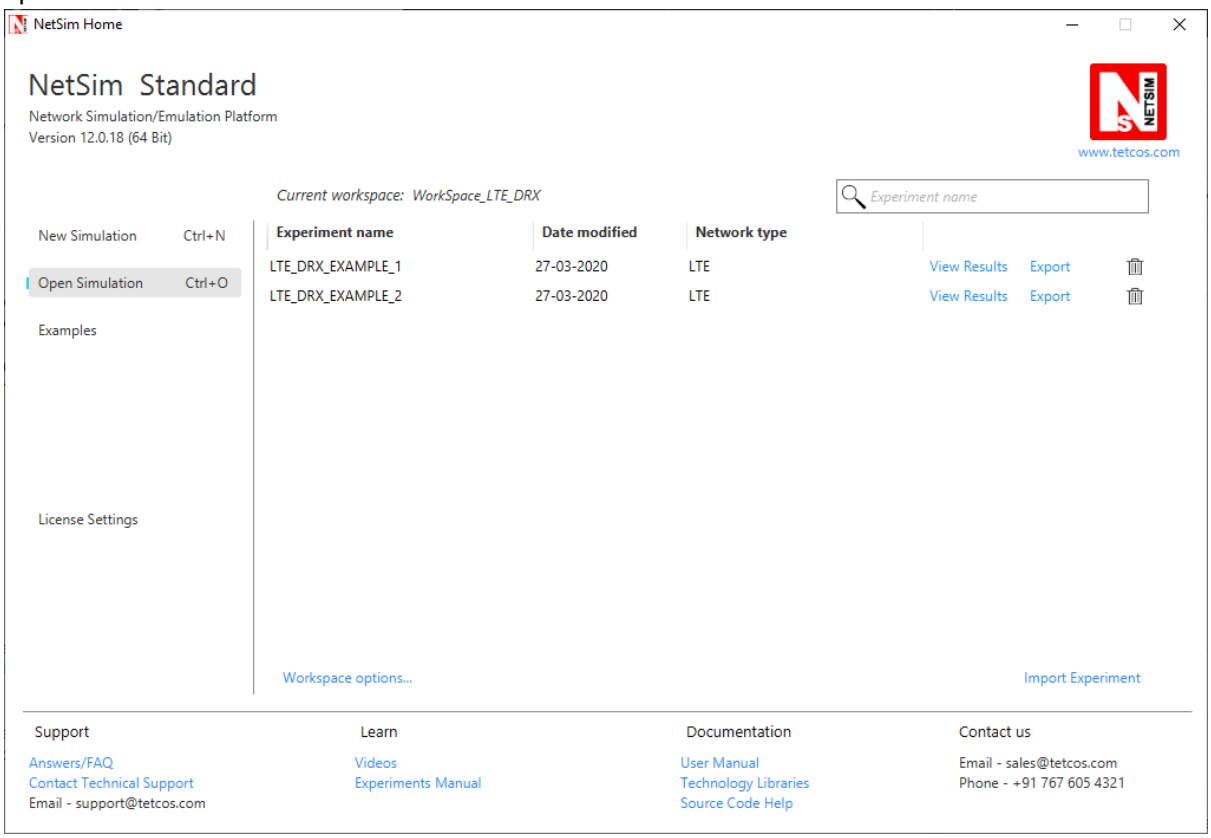
Other tools such as GitHub Desktop, SVN Client, Sourcetree, Git from the command line, or any client you like to clone the Git repository.

After cloning the workspace locally, you can import the WorkSpace_LTE_DRX into NetSim by going to Open Simulation -> Workspace Options -> More Options and clicking on the Import button.

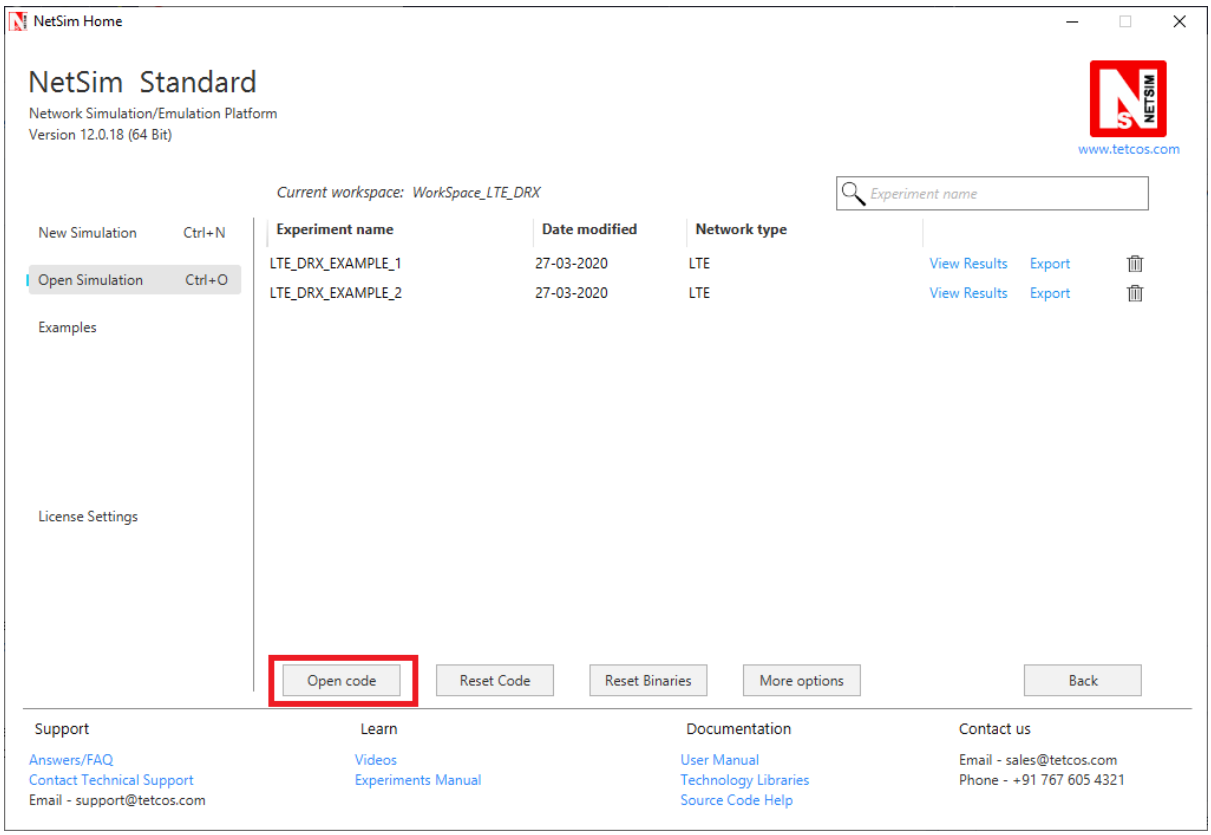
While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.



The imported workspace consists of two examples which can be opened from the Open Simulation option as shown below:

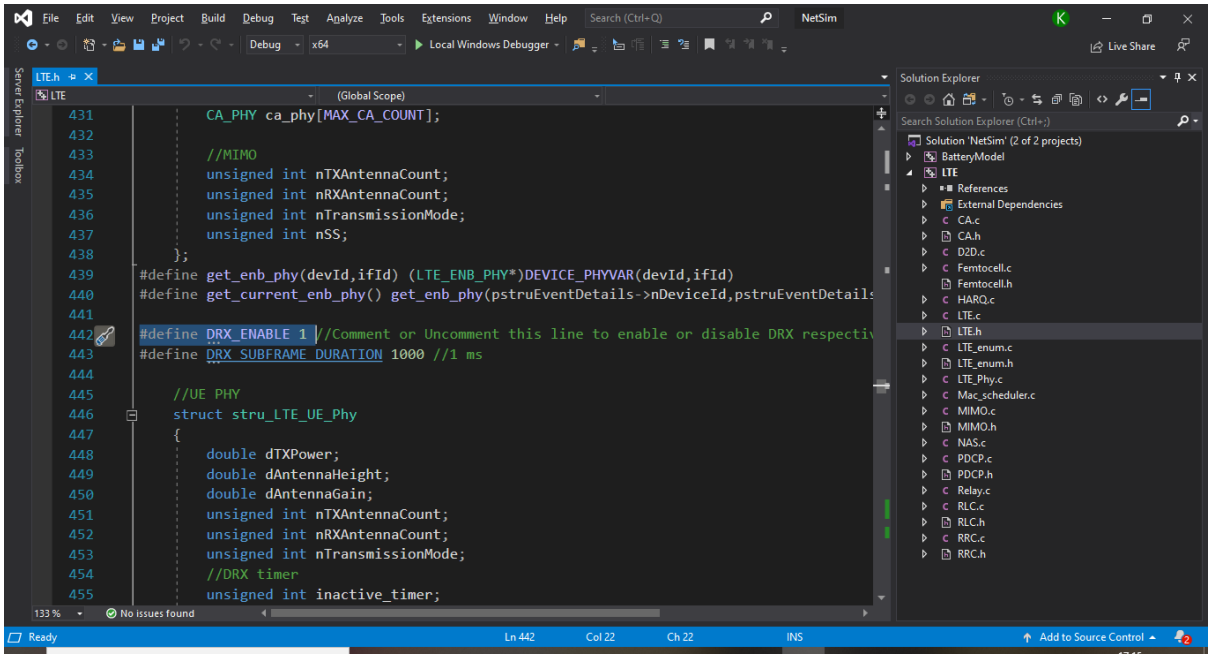


The source codes modified for this project can be accessed by going to Open Simulation -> Workspace Options and clicking on the Open Code button as shown below:

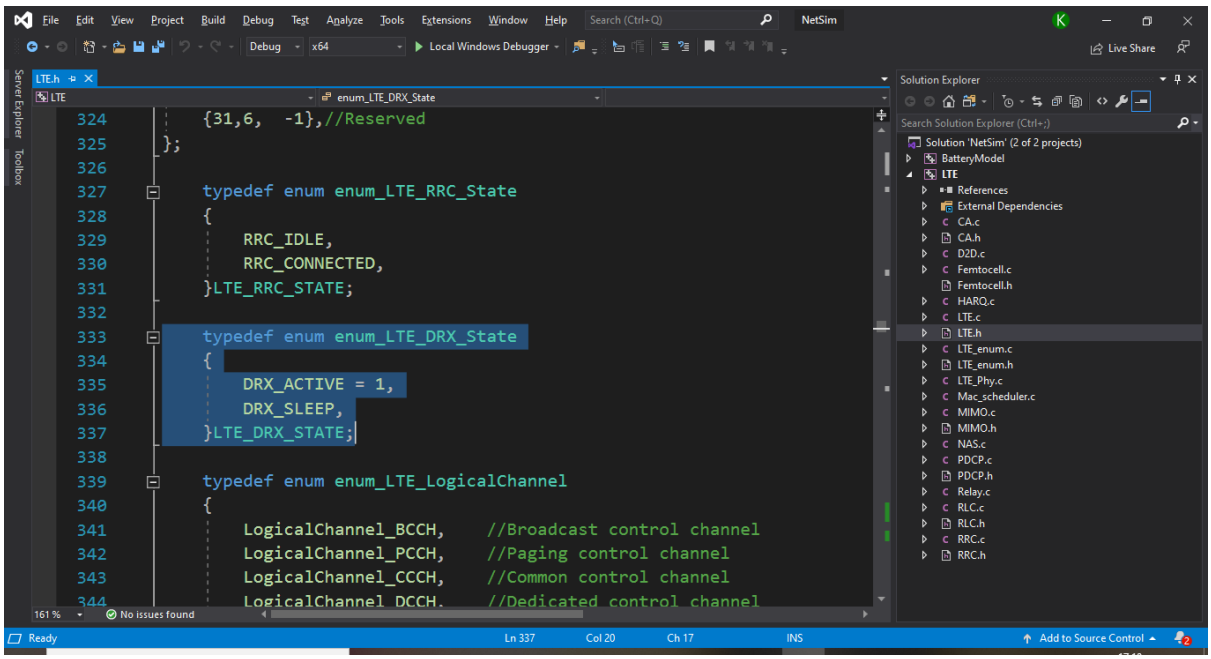


The files **LTE.c, LTE.h**, **LTE_enum.h**, **Mac_scheduler.c** and **RRC.c** which are part of the LTE project were modified for this implementation.

The line **#define DRX_ENABLE 1** in the **LTE.h** file can be commented to run simulations without DRX and get the energy consumption of devices in the results dashboard.
The same line can be uncommented to run simulations with DRX mode enabled for the UE's and get the energy consumption of the devices in the results dashboard.



The DRX states are defined in the LTE.h file as shown below:



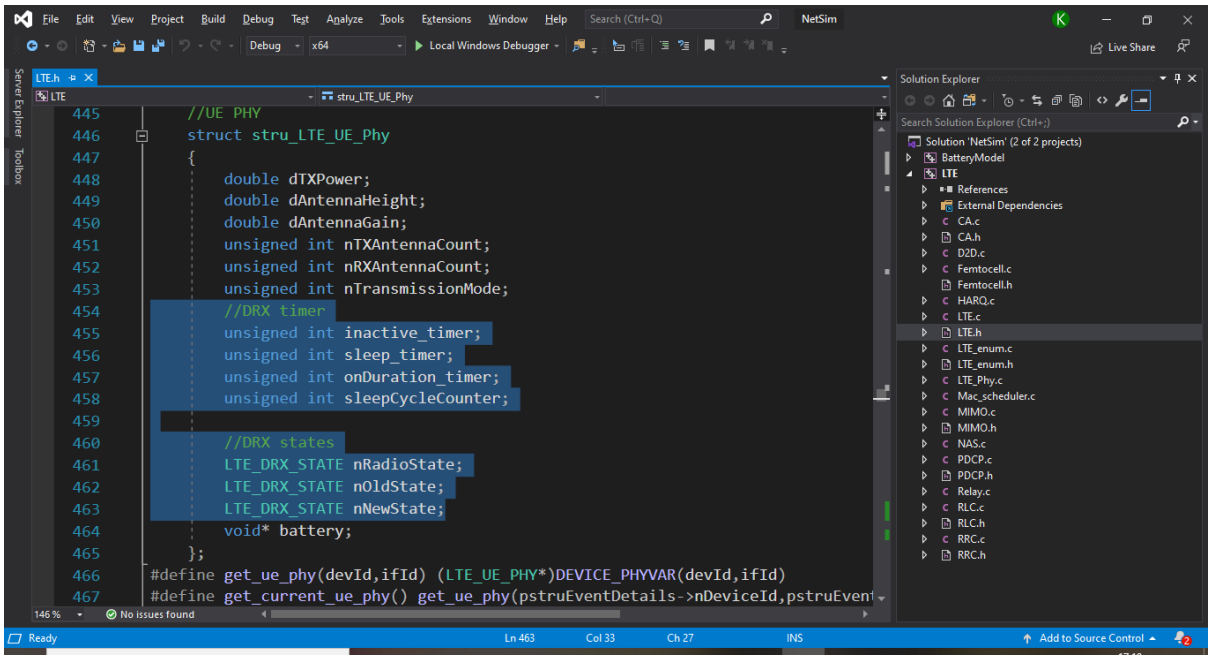
```
    {31,6,  -1},//Reserved
};

typedef enum enum_LTE_RRC_State
{
    RRC_IDLE,
    RRC_CONNECTED,
}LTE_RRC_STATE;

typedef enum enum_LTE_DRX_State
{
    DRX_ACTIVE = 1,
    DRX_SLEEP,
}LTE_DRX_STATE;

typedef enum enum_LTE_LogicalChannel
{
    LogicalChannel_BCCH,    //Broadcast control channel
    LogicalChannel_PCCH,    //Paging control channel
    LogicalChannel_CCCH,    //Common control channel
    LogicalChannel_DCCH,    //Dedicated control channel
```

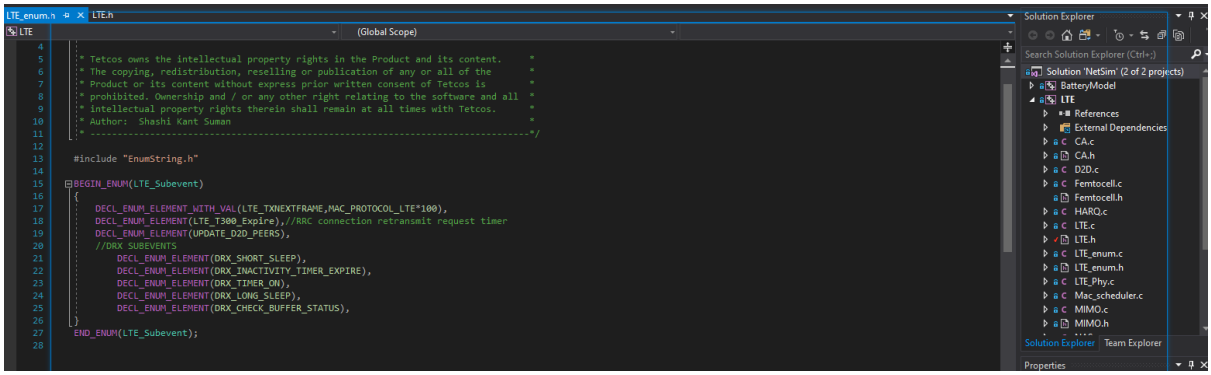
DRX timer and radio state related variables are added to the UE physical layer variable structure as shown below:



```
//UE PHY
struct stru_LTE_UE_Phy
{
    double dTXPower;
    double dAntennaHeight;
    double dAntennaGain;
    unsigned int nTXAntennaCount;
    unsigned int nRXAntennaCount;
    unsigned int nTransmissionMode;
    //DRX timer
    unsigned int inactive_timer;
    unsigned int sleep_timer;
    unsigned int onDuration_timer;
    unsigned int sleepCycleCounter;

    //DRX states
    LTE_DRX_STATE nRadioState;
    LTE_DRX_STATE nOldState;
    LTE_DRX_STATE nNewState;
    void* battery;
};
#define get_ue_phy(devId,ifId) (LTE_UE_PHY*)DEVICE_PHYVAR(devId,ifId)
#define get_current_ue_phy() get_ue_phy(pstruEventDetails->nDeviceId,pstruEvent
```

The various timer events related to DRX states and timers are declared in the LTE_enum.h file as shown below:

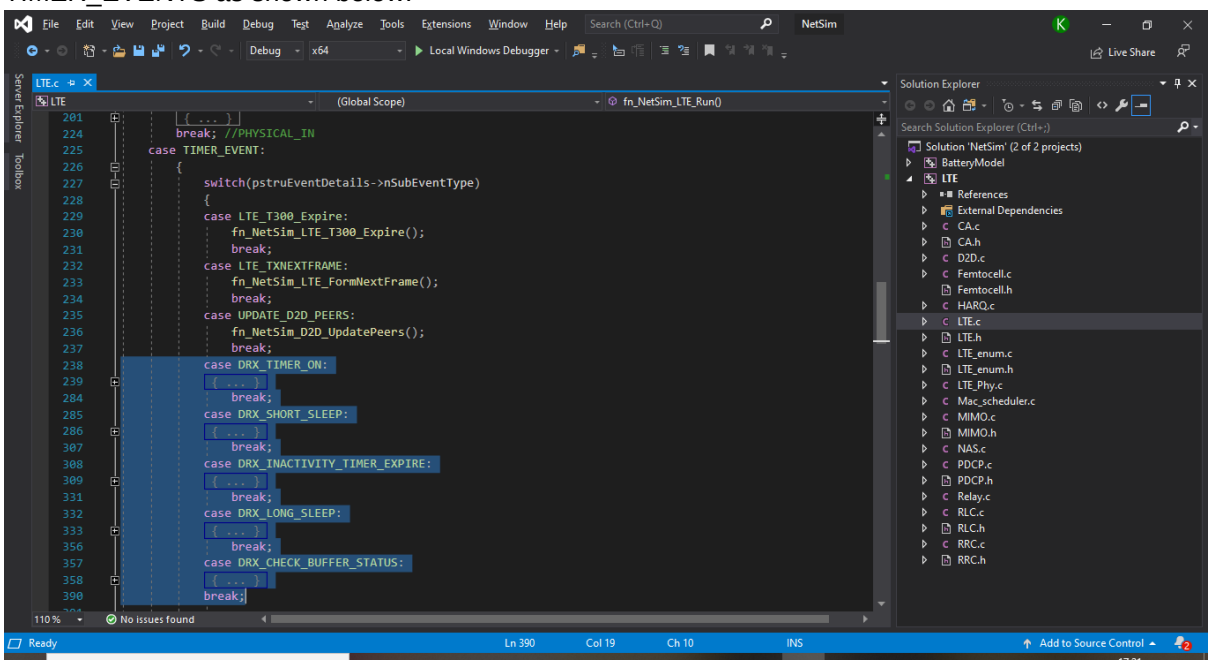


```
/* Tetcos owns the intellectual property rights in the Product and its content.    *
* The copying, redistribution, reselling or publication of any or all of the     *
* Product or its content without express prior written consent of Tetcos is      *
* prohibited. Ownership and / or any other right relating to the software and all *
* intellectual property rights therein shall remain at all times with Tetcos.    *
* Author:  Shashi Kant Suman                                                     *
*--------------------------------------------------------------------------------*/

#include "EnumString.h"

BEGIN_ENUM(LTE_Subevent)
{
    DECL_ENUM_ELEMENT_WITH_VAL(LTE_TXNEXTFRAME,MAC_PROTOCOL_LTE*100),
    DECL_ENUM_ELEMENT(LTE_T300_Expire),//RRC connection retransmit request timer
    DECL_ENUM_ELEMENT(UPDATE_D2D_PEERS),
    //DRX SUBEVENTS
        DECL_ENUM_ELEMENT(DRX_SHORT_SLEEP),
        DECL_ENUM_ELEMENT(DRX_INACTIVITY_TIMER_EXPIRE),
        DECL_ENUM_ELEMENT(DRX_TIMER_ON),
        DECL_ENUM_ELEMENT(DRX_LONG_SLEEP),
        DECL_ENUM_ELEMENT(DRX_CHECK_BUFFER_STATUS),
}
END_ENUM(LTE_Subevent);
```

The timer events related to DRX are defined in the funciton fn_NetSim_LTE_Run() under TIMER_EVENTS as shown below:



Some more sections of the LTE.c and RRC.c source codes were modified to add battery model for the UE's and to change radio states of the UE's periodically.

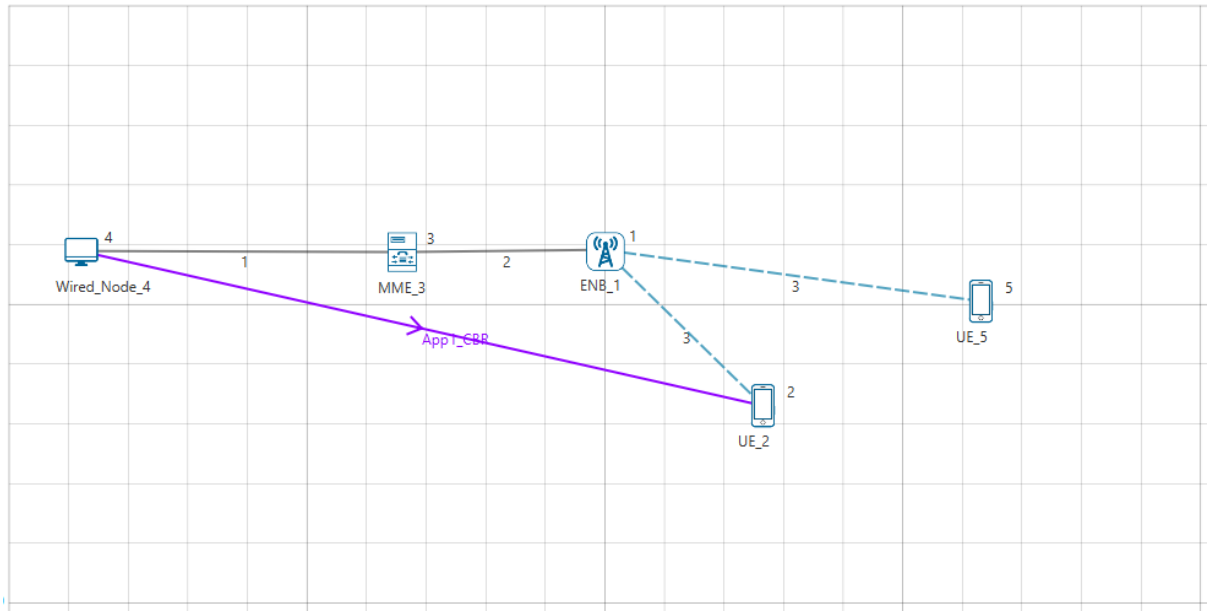
The source codes of the MAC_Scheduler.c file were modified to check whether the UE is in sleep state before adding packets to the downlink queue.

Simulation Results and Analysis:

In Visual Studio, set to platform to win32 or x64 based on the build of NetSim(32-bit or 64-bit) that you have installed.

Right click on the LTE project and select rebuild to build the source codes.

The LTE_DRX_EXAMPLE_1 consists of two UE's connected to a ENB out of which one UE performs download as shown below:



Upon running the simulations with and without DRX enable, we get the following energy consumption metrics:

With DRX disabled:

Battery model_Table

Battery model Detailed View

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Active energy(mJ)	Sleep energy(mJ)
UE_2	129600.000000	122.387760	129477.612240	122.387760	0.000000
UE_5	129600.000000	122.387760	129477.612240	122.387760	0.000000

With DRX enabled:

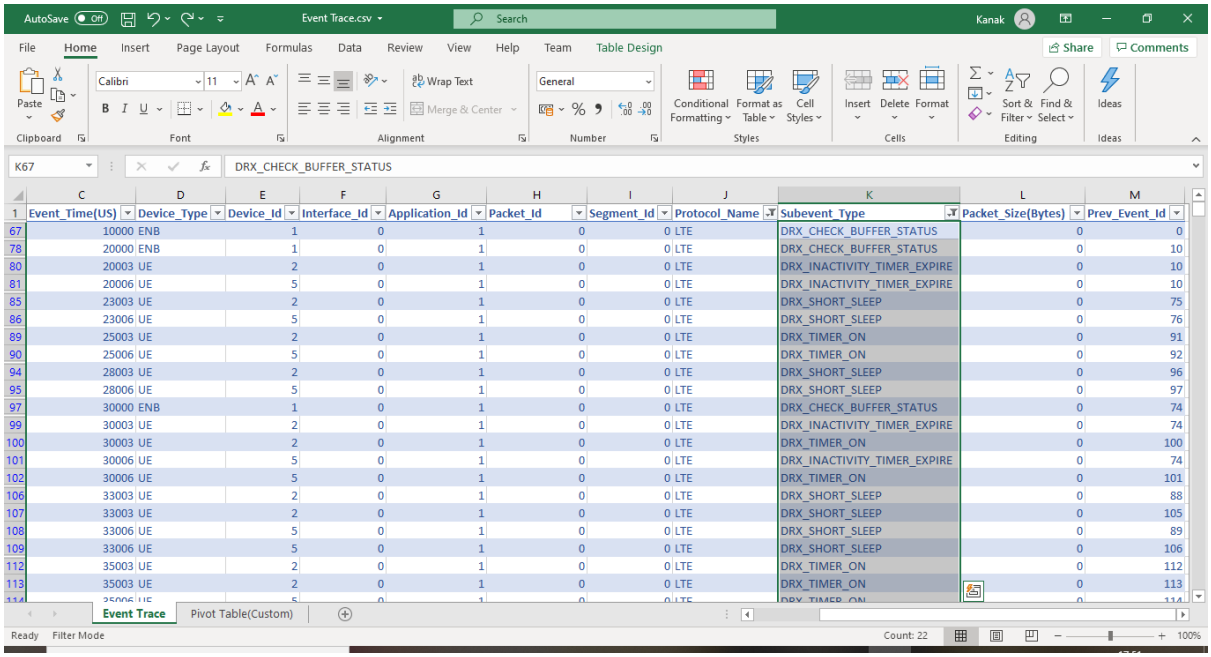
Battery model_Table

Battery model Detailed View

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Active energy(mJ)	Sleep energy(mJ)
UE_2	129600.000000	61.108582	129538.891418	18.298837	42.809745
UE_5	129600.000000	50.416603	129549.583397	0.122473	50.294130

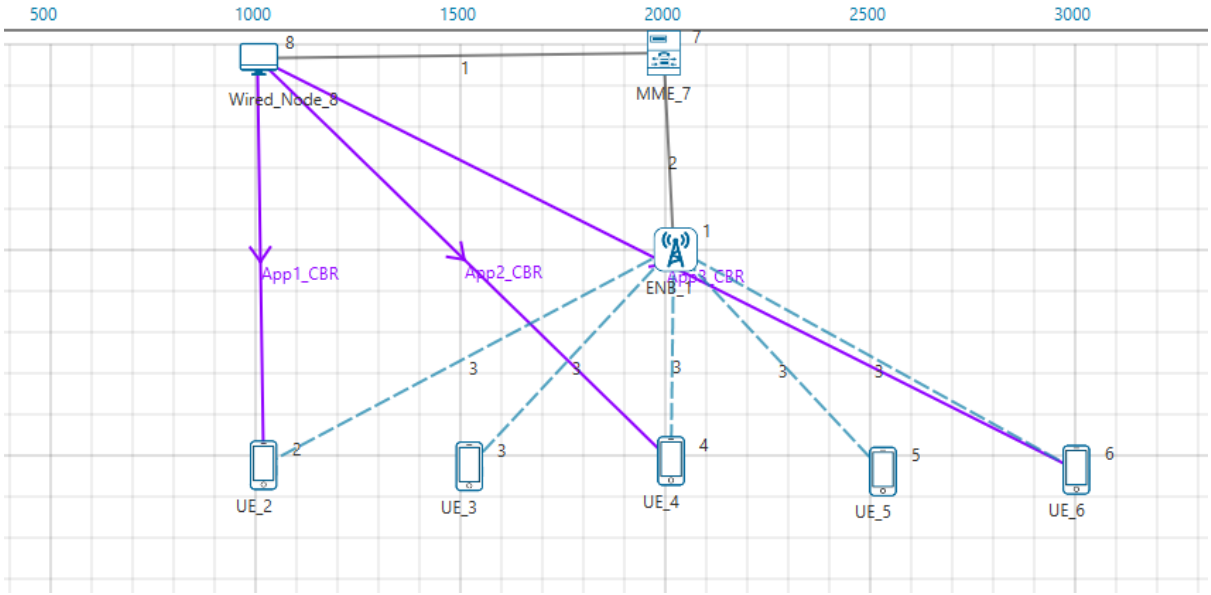
When DRX is enabled, nodes switch to sleep state due to which Sleep energy consumption can be found in the above table. Energy consumed by the devices are also lesser in case of DRX enabled since energy consumed in sleep mode is lesser than that of active mode.

The events related to DRX can be found in the Even Trace log file as shown below:



Event_Time(US)	Device_Type	Device_Id	Interface_Id	Application_Id	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size(Bytes)	Prev_Event_Id
10000	ENB	1	0	1	0	0	LTE	DRX_CHECK_BUFFER_STATUS	0	0
20000	ENB	1	0	1	0	0	LTE	DRX_CHECK_BUFFER_STATUS	0	10
20003	UE	2	0	1	0	0	LTE	DRX_INACTIVITY_TIMER_EXPIRE	0	10
20006	UE	5	0	1	0	0	LTE	DRX_INACTIVITY_TIMER_EXPIRE	0	10
23003	UE	2	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	75
23006	UE	5	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	76
25003	UE	2	0	1	0	0	LTE	DRX_TIMER_ON	0	91
25006	UE	5	0	1	0	0	LTE	DRX_TIMER_ON	0	92
28003	UE	2	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	96
28006	UE	5	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	97
30000	ENB	1	0	1	0	0	LTE	DRX_CHECK_BUFFER_STATUS	0	74
30003	UE	2	0	1	0	0	LTE	DRX_INACTIVITY_TIMER_EXPIRE	0	74
30003	UE	2	0	1	0	0	LTE	DRX_TIMER_ON	0	100
30006	UE	5	0	1	0	0	LTE	DRX_INACTIVITY_TIMER_EXPIRE	0	74
30006	UE	5	0	1	0	0	LTE	DRX_TIMER_ON	0	101
33003	UE	2	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	88
33003	UE	2	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	105
33006	UE	5	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	89
33006	UE	5	0	1	0	0	LTE	DRX_SHORT_SLEEP	0	106
35003	UE	2	0	1	0	0	LTE	DRX_TIMER_ON	0	112
35003	UE	2	0	1	0	0	LTE	DRX_TIMER_ON	0	113

The LTE_DRX_EXAMPLE_2 consists of five UE's connected to a ENB out of which three UE's performs download as shown below:



Upon running the simulations with and without DRX enable, we get the following energy consumption metrics

With DRX disabled:

Battery model_Table

Battery model Detailed View

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Active energy(mJ)	Sleep energy(mJ)
UE_2	129600.000000	122.387760	129477.612240	122.387760	0.000000
UE_3	129600.000000	122.387760	129477.612240	122.387760	0.000000
UE_4	129600.000000	122.387760	129477.612240	122.387760	0.000000
UE_5	129600.000000	122.387760	129477.612240	122.387760	0.000000
UE_6	129600.000000	122.387760	129477.612240	122.387760	0.000000

With DRX enabled:

Battery model_Table

Battery model Detailed View

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Active energy(mJ)	Sleep energy(mJ)
UE_2	129600.000000	61.108582	129538.891418	18.298837	42.809745
UE_3	129600.000000	50.416668	129549.583332	0.122584	50.294084
UE_4	129600.000000	61.108603	129538.891397	18.298873	42.809730
UE_5	129600.000000	50.416646	129549.583354	0.122547	50.294100
UE_6	129600.000000	61.108625	129538.891375	18.298910	42.809715

When DRX is enabled, nodes switch to sleep state due to which Sleep energy consumption can be found in the above table. Energy consumed by the devices are also lesser in case of DRX enabled since energy consumed in sleep mode is lesser than that of active mode.

The events related to DRX can be found in the Even Trace log file as shown below: