MERIF Workshop ORBIT/COSMOS Tutorial Session

For both the tutorials you are assigned a group color. As you work through the tutorial please replace all references to <color> in the commands shown below with your assigned color. Please be mindful of dashes :)
For example if your color is **cyan**, then replace

```
omf stat -t system:topo:group-<color>
```

with

```
omf stat -t system:topo:group-cyan
```

Tutorial 1

For the first tutorial, the wireless interface card on two nodes will be configured and a couple of applications will be launched to generate and receive traffic. The URL for this tutorial can be found on https://www.orbit-lab.org/wiki/Tutorials/a0Basic/Tutorial2 .

The tutorial script's interface has been slightly modified to accommodate MERIF tutorial groups so use the commands shown below and the details on the URL as a reference for the OEDL script break down.

OMF commands to configure nodes

We'll use OMF commands

1. to check status of nodes assigned to you
2. get FQDN of the nodes
3. load images, turn on/off the nodes.

Start by checking the current status of your nodes.

```
console> omf stat -t system:topo:group-<color>
```

You should see two nodes in your topology. This first tutorial will use both nodes. Load the disk OS image onto your nodes and then turn them on. After each omf command finishes wait about 10 seconds.

```
console> omf load -t system:topo:group-<color> -i merif-<color>.ndz
```

```
console> omf tell -a on -t system:topo:group-<color>
```

Wait a few moments for the node to come up. This process is similar to a laptop booting up it will go thru BIOS checks and load the operating system before giving you a login prompt.

Let's browse around inside a node. Get your nodes (X,Y) coordinates from the previous 'omf stat' command. For the commands below you will need to replace the nodeX-Y with your node coordinates.

Log into one of your node and have a quick look around. From inside the node, run a few example commands below.

```
console> ssh root@nodeX-Y

node> ls

node> ifconfig -a

node> lsb_release -a

node> uname -a

node> exit
```

Run the tutorial 1 experiment

Once you have exit out of the node and back at your console, you can download the experiment script to your local directory for viewing pleasure.

```
console> wget https://www.orbit-lab.org/raw-attachment/wiki/Tutorials/a0Basic/Tutorial2/hello-world-wireless.rb

console> cat hello-world-wireless.rb
```

Run the experiment script using the group color assigned to you. The script will use one node as a receiver and the other as transmitter.

```
console> omf exec test:exp:tutorial:hello-world-wireless.rb -- --group <color>
```

The output of the script execution should display an "Experiment ID" at the very bottom.

```
INFO run: Experiment default_slice-2019-05-24t18.38.07.840-04.00 finished after 0:26
```

The "default_slice_*" is your experiment ID output located in /tmp directory.

To view the details of this experimental run

```
console> cat /tmp/default_slice-2019-05-24t18.38.07.840-04.00.log
```

This was a simple demonstration of running two applications on just two nodes to send and receive traffic but can be scaled to simulate an experiment with larger set of nodes and other conditions.

At the end of a session it's always a good idea turn off your nodes.

```
console> omf tell -a offh -t system:topo:group-<color>
```

Tutorial 2

In this part of the tutorial you'll use the USRP on the node to collect samples from the radio and store them into an OML database. Then read out the samples from the database and plot the spectrum. For reference the details on this tutorial can be found on https://wiki.cosmos-lab.org/wiki/tutorials/measurement_tool .

You can download the OEDL script to your console.

```
console> wget https://wiki.cosmos-lab.org/raw-attachment/wiki/tutorials/measurement_tool/spectrum-demo.rb
```

Run the experiment

If you *need* you can reload your nodes with the disk OS image; otherwise skip this step.

```
console> omf load -t system:topo:group-<color> -i merif-<color>.ndz
```

Make sure your nodes are turned on. Use the 'omf tell' to switch the nodes back on. This part of the tutorial will only use a single node. When running the OEDL script, the output will show which node is being used as a receiver.

To run the OEDL script enter the following command at the console. (No dash between group and color.)

```
console> omf exec test:exp:tutorial:spectrum-demo -- --group <color>
```

The script will execute an application (/root/RX_MULTI_RECEIVE/rx_multi_receive) inside the node. The OEDL script configures the application to send post processed samples to the an OML server for storage.

Your experiment script was configured so that the OML server writes into /var/lib/oml2/<color>.sq3 on the console. The command line program *sqlite3* can be used to manually get information out of the database. Complete details on using sqlite3 can be found on https://www.sqlite.org/cli.html.

The .table command will list all the tables in the database. All the post processed data was written in the table _mp__<color>.

```
console> sqlite3 /var/lib/oml2/<color>.sq3 '.table'

console> sqlite3 /var/lib/oml2/<color>.sq3 '.schema'

console> sqlite3 /var/lib/oml2/<color>.sq3 '.dump'
```

Let's recover the the post processed samples (in this case FFT bins) from the database.

Download the source file for parsing from the tutorial site and extract it to your console. Change directory into it and compile the parser.

```
console> wget https://wiki.cosmos-lab.org/raw-attachment/wiki/tutorials/measurement_tool/rx_multi_receive.tar.gz

console> tar -zxvf rx_multi_receive.tar.gz

console> cd RX_MULTI_RECEIVE/

console> make
```

Do a directory listing and you should see an application called 'db_parse'. Run the parsing application with the following arguments.

```
console> ./db_parse --file /var/lib/oml2/<color>.sq3 --showtables
```

This should show you a list of tables present in the database file. The FFT bins are located in the tables named: _mp__<color>. Next issue the following command to extract the FFT bins into a binary file. For convenience the command is also printed in the last line of the output from last command. :)

```
console> ./db_parse --file /var/lib/oml2/<color>.sq3 --blob2bin ch0_data.bin --ch 0 --table  _mp__<color> --key Bins
```

This should create a binary file 'ch0_data.bin' in the same directory. Now run octave from the console and enter the following commands to get a spectrum image.

```
console> octave
octave> mag0 = fReadBinaryFile('ch0_data.bin','single');
octave> mag0 = reshape(mag0, 1024, length(mag0)/1024);
octave> image(mag0 * 1000);
```