

# COSMOS Operation Instruction

Yuning Zhang

January 2022

[Reservation for sb1 \(COSMOS\)](#) and use the [Time Zone Sheet](#) to reserve the correct slots.

Reservation website login info: wides2021, Wides2021Brian

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Useful Links</b>	<b>3</b>
<b>3</b>	<b>SSH Configuration</b>	<b>3</b>
<b>4</b>	<b>Testbed Topology</b>	<b>4</b>
<b>5</b>	<b>X-Y Table</b>	<b>7</b>
5.1	Information . . . . .	7
5.2	Usage . . . . .	8
5.3	Examples . . . . .	8
5.4	Video Monitoring . . . . .	9
<b>6</b>	<b>Switch</b>	<b>10</b>
6.1	Usage . . . . .	10
6.2	Connection . . . . .	10
<b>7</b>	<b>Hardware</b>	<b>11</b>
7.1	Siver's 60GHz array . . . . .	11
<b>8</b>	<b>Tutorial: 60GHz RFSoc</b>	<b>11</b>
8.1	Preparation . . . . .	11

8.1.1	System Check and Reset . . . . .	11
8.1.2	Load Image . . . . .	13
8.1.3	Connect the Switch . . . . .	14
8.2	RFSoc Setup . . . . .	14
8.3	Basic Demos . . . . .	15
8.4	Upload Data . . . . .	16
8.5	Git . . . . .	16
8.6	Save Images . . . . .	16
<b>9</b>	<b>Tutorial: 60GHz USRP</b>	<b>16</b>
9.1	Preparation . . . . .	16
9.1.1	System Check and Reset . . . . .	17
9.1.2	Load Image and update Git . . . . .	18
9.1.3	Turn on Nodes . . . . .	18
9.1.4	Connect the Switch . . . . .	19
9.2	USRP Setup . . . . .	19
9.2.1	GUI and video preparation . . . . .	19
9.2.2	Run Measurement . . . . .	20
9.3	Attention . . . . .	22
9.3.1	USRP Identification . . . . .	22
9.3.2	Daughterboards . . . . .	23
9.3.3	Subdevice . . . . .	23
9.4	Upload Data . . . . .	24
9.5	Git . . . . .	24
9.6	Save Images . . . . .	24
<b>10</b>	<b>Measurement Required Devices</b>	<b>24</b>
10.1	XY table . . . . .	24
<b>11</b>	<b>Error Records</b>	<b>27</b>
11.1	Timeout when loading the image to servers . . . . .	27
11.1.1	Description . . . . .	27
11.1.2	Solution . . . . .	27
11.2	Node Not Available Error . . . . .	28

## 1 Introduction

COSMOS: Cloud-Enhanced Open Software Defined Mobile Wireless Testbed for City-Scale Deployment. It’s awarded by NSF under the Platforms for Advanced Wireless Research (PAWR) program. [Click](#).

All information presented as a slide

## 2 Useful Links

- [Intro](#): COSMOS Testbed Selected as One of Nation’s First FCC Innovation Zones
- [Slide](#): IEEE FNI Testbed WG Meeting.
- [Slide](#): COSMOS Project Overview
- [Paper](#): Challenge: COSMOS: A City-Scale Programmable Testbed for Experimentation with Advanced Wireless
- [Paper](#): Programmable and Open-Access Millimeter-Wave Radios in the PAWR COSMOS Testbed
- [Video](#): COSMOS Demonstrates New Open-Source, Open-Access mmWave Beam Tracking Testbed
- [Code](#): GitHub source
- [mmwaveSB1](#)
- [COSMOS Wiki](#)

## 3 SSH Configuration

Download PuTTY and Pageant from the PuTTY website, and configure them as below:

<b>Host Name/IP Address</b>	console.sb1.cosmos-lab.org
<b>Type</b>	SSH
<b>Private Key</b>	SB1 Private Key.ppk
<b>Login Name</b>	wides2021
<b>PW</b>	WIDES2021SB1

Then here are some optional configuration with the terminal:

<b>Enable X11 forwarding</b>	Check
<b>X display location</b>	localhost:0
<b>Cursor Appearance</b>	Block
<b>Font</b>	Consolas, 18-point
<b>Font Quality</b>	Default
<b>Windows Size</b>	150 columns, 200 rows
<b>Lines of scroll-back</b>	2000

## 4 Testbed Topology

The topology of the sandbox 1 ([COSMOS Testbed 1](#)) contains multiple sections: Rooftop Interdigital Edgeline, Main Rooftop SDR Deployment, Benchtop mmWave SDR Setup, Core Servers and Networking, and Benchtop Interdigital 5G NR Performance System. The only section that we (WiDeS) are interested in is the Benchtop mmWave SDR Setup, as the [Figure 1](#) shows.

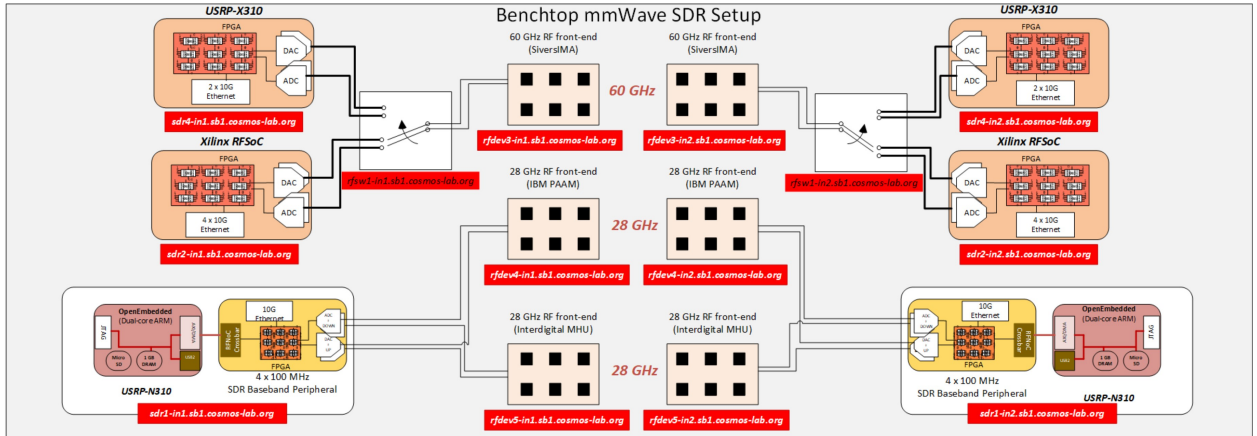


Figure 1: Benchtop Topology

The nodes have particular names that can be recognized by the server, here is the table of the nodes from the benchtop section and some other accessory nodes (servers and such). The SDR name usually will followed by “.sb1.cosmos-lab.org”, but can be omitted with the bash commands.

Check the equipment lists by using the command below to see if all hardware you need is listed:

```
yourusername@console:~$ omf stat -t system:topo:allres
```

### Sandbox 1 resources

Node Name	SDR Name(.sb1)	Frequency
Hand-held USRP-E312	mob1-1 (10.37.10.1)	NaN
small Intel NUC & B205-mini	mob2-1 (10.37.11.1)	NaN
big Intel NUC & B210	mob3-1 (10.37.12.1)	NaN
EdgeLink Device 1	rfdev1-in1 (10.37.4.1)	60GHz PtP Link
EdgeLink Device 2	rfdev1-in2 (10.37.4.2)	60GHz PtP Link
5G NR BS	rfdev2-in1 (10.37.5.1)	28GHz
5G NR UE	rfdev2-in2 (10.37.5.2)	28GHz
60GHz RF Front-end: Silver array (left, SN0240)	rfdev3-in1	60GHz
60GHz RF Front-end: Silver array (right, SN0243)	rfdev3-in2	60GHz
28GHz RF Front-end: IBM PAAM (left)	rfdev4-in1	28GHz
28GHz RF Front-end: IBM PAAM (right)	rfdev4-in2	28GHz
28GHz RF Front-end: Interdigital MHU (left)	rfdev5-in1	28GHz
28GHz RF Front-end: Interdigital MHU (right)	rfdev5-in2	28GHz
USRP-N310 + SDR Baseband Peripheral (left)	sdr1-in1 (10.37.6.1)	28GHz
USRP-N310 + SDR Baseband Peripheral (right)	sdr1-in2 (10.37.6.2)	28GHz
Not show	sdr1-in3	NaN
USRP-N310 in medium node	sdr1-md1 (10.37.3.1)	NaN
USRP-N310 (sector 1) in large node	sdr1-s1-lg1 (10.37.2.1)	NaN
Xilinx RFSoc (left)	sdr2-in1 (10.37.6.3)	NaN
Xilinx RFSoc (right)	sdr2-in2 (10.37.6.4)	NaN
USRP-2974 in medium node	sdr2-md1 (10.37.3.2)	NaN
USRP-2974 (sector 1) in large node	sdr2-s1-lg1 (10.37.2.2)	NaN
Not show	sdr3-in1	NaN
USRP-X310 (left)	sdr4-in1 (10.38.14.1)	NaN
USRP-X310 (right)	sdr4-in2 (10.38.14.2)	NaN
RF Switch (left)	rfsw1-in1	sdr2 & sdr 4
RF Switch (right)	rfsw1-in2	sdr2 & sdr 4
Sub-Servers:1-2 (mmWave)	srv1-in1/2 (10.37.1.3/4)	N/A
Sub-Servers:3 (mobile)	srv1-in3 (10.37.1.5)	N/A
Core-Servers:1-2	srv1/2-lg1 (10.37.1.1/2)	N/A

## Sandbox 2 resources

Node Name	SDR Name(.sb2)	Frequency
Xilinx RFSoc (left)	rfdev1-in1 (10.116.4.1)	NaN
Xilinx RFSoc (right)	rfdev1-in2 (10.116.4.2)	NaN
28GHz RF Front-end: IBM PAAM (left)	rfdev2-in1 (10.116.5.1)	28GHz
28GHz RF Front-end: IBM PAAM (right)	rfdev2-in2 (10.116.5.2)	28GHz
FD?	rfdev3-in1	not listed
FD?	rfdev3-in2	not listed
USRP-N310 in medium node	sdr1-md1 (10.116.3.1)	NaN
USRP-2974 (Krypton) in medium node	sdr2-md1 (10.116.3.2)	NaN
USRP-N310 in large node	sdr1-s1-lg1 (10.116.2.1)	NaN
USRP-2974 (Krypton) in large node	sdr2-s1-lg1 (10.116.2.2)	NaN
RF Switch (large node)	rfsw-s1-lg1	<a href="#">figure</a>
RF Switch (medium node)	rfsw-md1	<a href="#">figure</a>
Core-Servers:1-2	srv1/2-lg1 (10.116.1.1/2)	N/A

## Bed resources

Node Name	SDR Name(.bed)	Frequency
Medium Note at 1st floor	sdr2-md1	<a href="#">120th street</a>
Medium Note at 2nd floor	sdr1-md2	<a href="#">Amsterdam Ave</a>
Medium Note at 2nd floor	sdr2-md2	<a href="#">Amsterdam Ave</a>
Big Node 1, sector 2, sdr1, at 18th floor	sdr1-s2-lg1	NaN
Big Node 1, sector 3, sdr1, at 18th floor	sdr1-s3-lg1	NaN
Big Node 1, sector 1, sdr2, at 18th floor	sdr2-s1-lg1	NaN
Big Node 1, sector 2, sdr2, at 18th floor	sdr2-s2-lg1	NaN
Big Node 1, sector 3, sdr2, at 18th floor	sdr2-s3-lg1	NaN
Core-server by all	srv1-co1	N/A
Core-server by all	srv2-co1	N/A
Server for 3 Big Nodes (18th floor)	srv1-lg1	N/A
Server for 3 Big Nodes (18th floor)	srv2-lg1	N/A
Server for 3 Big Nodes (18th floor)	srv3-lg1	N/A
Server for 3 Big Nodes (18th floor)	srv4-lg1	N/A

The switches and the X-Y table will not show up in the "omf stat" check list because they are not considered as "nodes". According to Jakub Kolodziejski, "*Nodes are devices that the user can control the power state of and interact with **directly** (e.g. SSH, serial-over-USB, etc). The RF switches and X-Y tables are not directly accessible by the user and are instead meant to be interacted with indirectly via out backend service APIs*". That's why you don't see the switches and the X-Y table in Figure 6a.

## 5 X-Y Table

### 5.1 Information

The [X-Y table setup](#) contains 2 tables, and each of them has 2 sliding tracks so that the arrays mounted on the tracks can be controlled to move to a particular position. The X-Y table cannot be accessed directly by the SSH command, instead, it requires the HTTP API to indirectly control, which locates at "[am1.cosmos-lab.org:5054/xy\\_table](http://am1.cosmos-lab.org:5054/xy_table)" You can access it within the console. The specific access will be discussed later.

The dimension of the X-Y tables are 1.3m x 1.3m, so the movement is limited within a 1.3m x 1.3m area. The antenna arrays may be rotated  $\pm 45^\circ$ . The centers of the tables are approximately 20m apart. The coordinate systems (including the origin point) has been flipped  $180^\circ$  between the two tables as shown by [Figure 2](#).

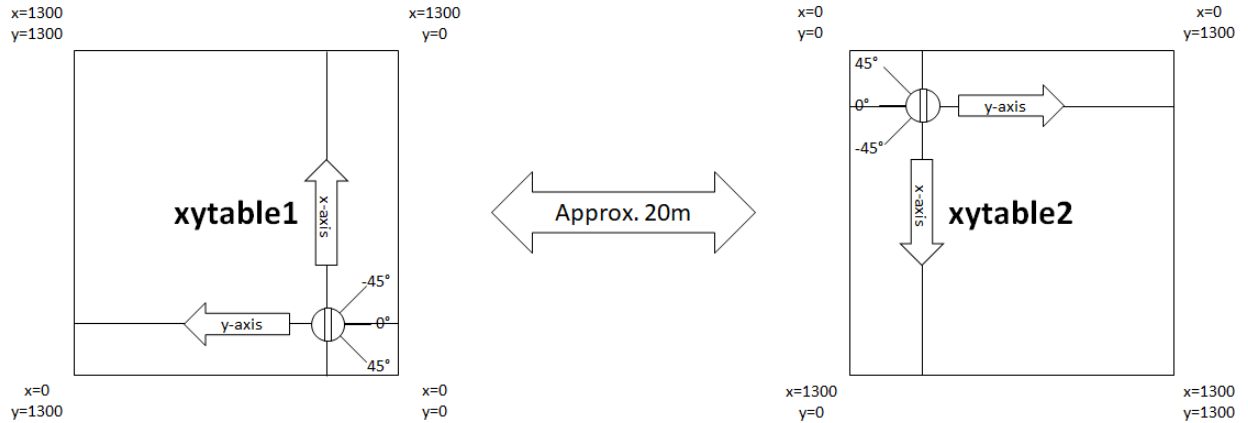


Figure 2: X-Y Table positions

The small circle in each table represents the array, the  $0^\circ$  direction is the array front side. So if you want to align two arrays without any rotation, the x-axis of one array and the x-axis of the other should be summed at 1300 (mm). The rotation positive direction is always defined as clockwise. They (COSMOS) can ensure[1] (they may update the Wiki) that the arrays are facing each other if the antennas are aligned on the x-axis and no rotation being involved.

## 5.2 Usage

All the available methods are accessible from the testbed console via an HTTP API. It can be interacted with via a command line tool such as "curl" or any script using a "REST client" library. According to Jakub Kolodziejewski, *"Since they are HTTP REST-like, you must use something like "curl" from the command line or if you have a script, then use the HTTP request library of your choice"*.

The particular way to use the "curl" command is:

```
curl "am1.cosmos-lab.org:5054/xy_table/<command>?<parameter1>=value&<parameter2>=value..."
```

Here are some of the APIs:

- **Status & Positions Check: status**

Parameter Name	Description	Required?	Valid Inputs
name	Comma between XY table FQDNs	Y	XY table FQDN list

Example for one valid input: "xytable1.sb1.cosmos-lab.org". FQDN: Fully Qualified Domain Name.

- **Movement: move\_to**

Parameter Name	Description	Required?	Valid Inputs
name	Comma between XY table FQDNs	Y	XY table FQDN list
x	X-axis coordinate (mm)	Y	0 to 1300
y	X-axis coordinate (mm)	Y	0 to 1300
angle	Targeted angle (deg)	Y	-45 to 45

- **Stop Movement: stop**

Parameter Name	Description	Required?	Valid Inputs
name	Comma between XY table FQDNs	Y	XY FQDN list

## 5.3 Examples

- **Check X-Y Table 1 Status**

```
curl "am1.cosmos-lab.org:5054/xy_table/status?name=xytable1.sb1.cosmos-lab.org"
```

If you want to check the status of both X-Y tables, just add a comma after the -lab.org (without space bar), and then add the name of the X-Y table 2.

- **Move X-Y Table**



```
curl "am1.cosmos-lab.org:5054/xy_table/move_to?name=
xytable1.sb1.cosmos-lab.org&x=500&y=30&angle=15"
```

From the observation, you may notice that there might be a constant shift between the current position/coordinate and the target position/coordinate. For example, Figure 3 shows the error of the XY table 1 between the current position and the target position. And the error might be changing from time to time.

```
wides2021@console:~$ curl "am1.cosmos-lab.org:5054/xy_table/status?name=xytable1.sb1.cosmos-lab.org"
<?xml version="1.0" encoding="UTF-8"?>
<response status="OK">
  <action service="xy_table" name="status">
    <xy_table xy_status="Idle" rotator_status="Holding" name="xytable1.sb1.cosmos-lab.org">
      <current_position x="1300.999" y="303.992" angle="-10.5"/>
      <target_position x="1300" y="303" angle="-10.0"/>
    </xy_table>
  </action>
</response>
```

Figure 3: X-Y Table Position Error

Note down both coordinates and their difference at every measurement, and try no to do measurement at both ends (0 or 1300). If the coordinate shift is not constant, then you can always re-compensate or re-adjust the target position to get the same **current position**, or same what? **Double check with COSMOS, which parameter should be used to achieve a repeatable measurement.**

## 5.4 Video Monitoring

Use the command below to access the sub-server for a better video streaming:

```
yourusername@console:~$ ssh -Y -c aes128ctr root@srv1-in1
```

```
yourusername@console:~$ ssh -Y -c aes128-ctr root@srv1-in1
```

Access it by (My own customized image):

```
root@srv1-in1:~$ cd mmwsdr/host/demos/basic/
```

```
root@srv1-in1:~/mmwsdr/host/demos/basic/# python video.py --node srv1-in1 --time 10
```

```
root@srv1-in1:~$ cd mmwsdr/host/demos/basic/
```

```
root@srv1-in1:~/mmwsdr/host/demos/basic/# python video.py --node srv1-in1 --time 10
```

For COSMOS default image, use:

```
root@srv1-in1:~$ cd mmwsdr/host/demos/basic/
```

```
root@srv1-in1:~/mmwsdr/host/demos/basic/# python video.py --node srv1-in1
```

```

root@srv1-in1:~$ cd mmwsdr/host/demos/basic/

root@srv1-in1:~/mmwsdr/host/demos/basic/# python video.py --node srv1-in1

root@srv1-in1:~/mmwsdr/host/demos/basic/# python3 video.py --node srv1-in1

```

## 6 Switch

[Switch](#) is another kind of node that will not be listed by the “omf” command.

### 6.1 Usage

- **Setup: set**

Parameter Name	Description	Required?	Valid Inputs
name	best not to put multiple switches	Y	switch FQDN list
switch	comma between sub-switch	Y	1 to 4
port	port on sub-switch	Y	1 or 2

Copy the command one line at a time, no space bar, and press enter after you paste all rows.

```

curl "am1.cosmos-lab.org:5054/rf_switch/set?name=rfs1.sb1.cosmos-lab.org,
rfs2.sb1.cosmos-lab.org&switch=1,2,3,4&port=2"

```

- **Check status: status**

Parameter Name	Description	Required?	Valid Inputs
name	Comma between RF Switch FQDNs	Y	RF Switch FQDN list

The usage is the same as the X-Y table status check.

```

curl "am1.cosmos-lab.org:5054/rf_switch/status?name=rfs1.sb1.cosmos-lab.org"

```

### 6.2 Connection

The switch 1 and 2 are connected between the 60GHz Sivers Array and either the Xilinx RFSoc or the USRP X310. Port 1 for all 4 sub-switches in both RF switches represent the USRP-X310, and port 2 for all 4 sub-switches in both RF switches connect to the RFSoc. For the RFSoc connection, the specific port connection is shown as Figure 4.

And the connection is:

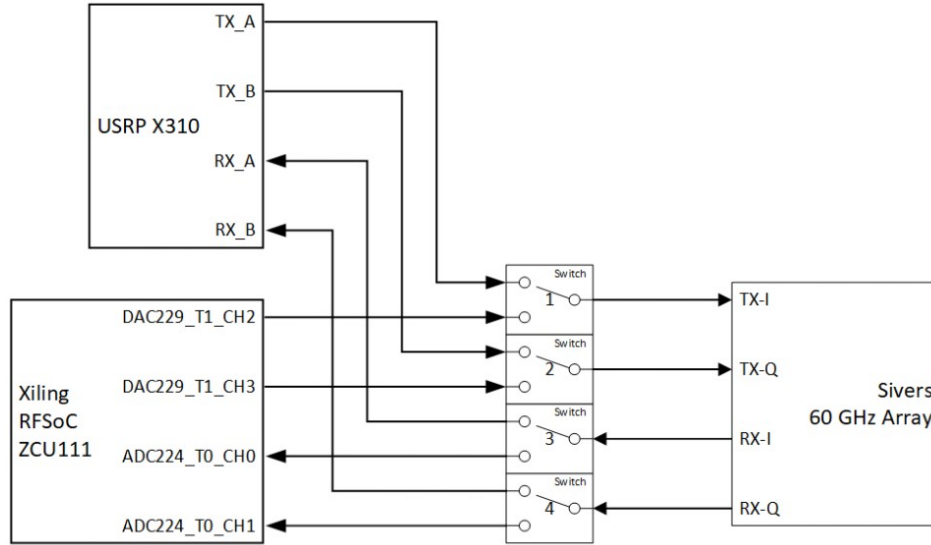


Figure 4: RF Switch Connection

Xilinx RFSoc	Sivers Array
DAC229_T1_CH2	TX-I
DAC229_T1_CH3	TX-Q
DAC229_T0_CH0	RX-I
DAC229_T0_CH1	RX-Q

## 7 Hardware

### 7.1 Siver's 60GHz array

The link of this product is [here](#). It is a 16\*4\*2 array.

## 8 Tutorial: 60GHz RFSoc

Follow the [tutorial](#) of the wiki.

### 8.1 Preparation

#### 8.1.1 System Check and Reset

When you access the server by SSH, you will have such an interface (example with SB1) as shown by Figure 5.

```

sb1.cosmos-lab.org - PuTTY
Using username "wides2021".
Authenticating with public key "rsa-key-20210407"
Passphrase for key "rsa-key-20210407":
Welcome to

COSMOS-LAB.ORG

Hostname      : console.sb1.cosmos-lab.org
Operating system : Ubuntu 16.04.7 LTS; Kernel: 4.15.0-142-generic; Arch: x86_64;
CPU           : 6 x Intel Xeon Processor (Skylake)
                1 socket(s) with 6 core(s) per socket and 1 thread(s) per core
Memory        : 3.9G
Uptime        : up 9 weeks, 4 days, 22 hours, 45 minutes
Users logged in : 0
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Mon Jan 17 20:24:07 2022 from your public IP Address
wides2021@console:~$

```

Figure 5: Resource being successfully reserved

First, you need to make sure all the nodes and devices of this reservation are turned off by the command:

```

yourusername@console:~$ omf tell -a offh -t system:topo:allres
yourusername@console:~$ omf stat -t system:topo:allres

```

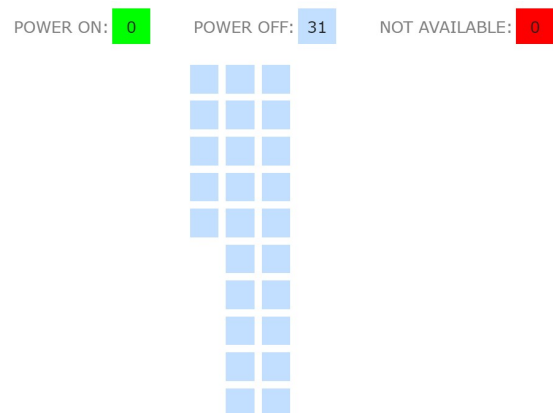
And what you should have is the Figure 6a. You may also double check if the power off operation is done correctly just by going to the [status](#) page, and check the node status as the Figure 6b shows.

```

Talking to the CMC service, please wait
-----
Node: mob1-1.sb1.cosmos-lab.org      Reply: OK
Node: mob2-1.sb1.cosmos-lab.org      Reply: OK
Node: mob2-2.sb1.cosmos-lab.org      Reply: OK
Node: mob3-1.sb1.cosmos-lab.org      Reply: OK
Node: rfdev1-1.sb1.cosmos-lab.org    Reply: OK
Node: rfdev1-2.sb1.cosmos-lab.org    Reply: OK
Node: rfdev2-in1.sb1.cosmos-lab.org  Reply: OK
Node: rfdev2-in2.sb1.cosmos-lab.org  Reply: OK
Node: rfdev3-in1.sb1.cosmos-lab.org  Reply: OK
Node: rfdev3-in2.sb1.cosmos-lab.org  Reply: OK
Node: rfdev4-in1.sb1.cosmos-lab.org  Reply: OK
Node: rfdev4-in2.sb1.cosmos-lab.org  Reply: OK
Node: rfdev5-in1.sb1.cosmos-lab.org  Reply: OK
Node: rfdev5-in2.sb1.cosmos-lab.org  Reply: OK
Node: sdr1-in1.sb1.cosmos-lab.org    Reply: OK
Node: sdr1-in2.sb1.cosmos-lab.org    Reply: OK
Node: sdr1-in3.sb1.cosmos-lab.org    Reply: OK
Node: sdr1-md1.sb1.cosmos-lab.org    Reply: OK
Node: sdr1-s1-lg1.sb1.cosmos-lab.org Reply: OK
Node: sdr2-in1.sb1.cosmos-lab.org    Reply: OK
Node: sdr2-in2.sb1.cosmos-lab.org    Reply: OK
Node: sdr2-md1.sb1.cosmos-lab.org    Reply: OK
Node: sdr2-s1-lg1.sb1.cosmos-lab.org Reply: OK
Node: sdr3-in1.sb1.cosmos-lab.org    Reply: OK
Node: sdr4-in1.sb1.cosmos-lab.org    Reply: OK
Node: sdr4-in2.sb1.cosmos-lab.org    Reply: OK
Node: srv1-in1.sb1.cosmos-lab.org    Reply: OK
Node: srv1-in2.sb1.cosmos-lab.org    Reply: OK
Node: srv1-in3.sb1.cosmos-lab.org    Reply: OK
Node: srv1-lg1.sb1.cosmos-lab.org    Reply: OK
Node: srv2-lg1.sb1.cosmos-lab.org    Reply: OK
-----

```

(a) Resource being successfully reserved



(b) Status when all nodes are powered off

Figure 6: Powered Off All Nodes

### 8.1.2 Load Image

When the system is powered off, you can **load the system image into the needed servers** by using the command “omf load” (check your correct image from the root directory: /export/omf-images-5.4):

```
yourusername@console:~$ omf load -i wides2021_rfsoc.ndz -t srv1-in1,srv1-in2
"rfsoc_sivers_sb1.ndz"
yourusername@console:~$ omf stat -t system:topo:allres
```

Sometimes there will be an error that described in Section 11.1, then just follow the corresponded solution.

Figure 7 is the screenshot that indicates the image has been loaded successfully.

If everything is correct, then what you will see is a list of Figure 6a, just the “Reply: OK” part will be replaced by “State: POWEROFF”. (Always remember such image-loading process must be done when all the nodes (or at least the servers) are powered off).

```
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout: 780
sec.
INFO exp: Progress(0/0/2): 10/10/10 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout:
70 sec.
INFO exp: Progress(0/0/2): 20/20/20 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout:
60 sec.
INFO exp: Progress(0/0/2): 30/30/30 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout:
50 sec.
INFO exp: Progress(0/0/2): 50/50/50 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout:
40 sec.
INFO exp: Progress(0/0/2): 60/60/60 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout:
30 sec.
INFO exp: Progress(0/0/2): 80/80/80 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout: 720 sec.
INFO exp: Progress(0/0/2): 90/90/90 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout: 710 sec.
INFO exp: Progress(1/0/2): 90/95/100 min(srv1-in1.sb1.cosmos-lab.org)/avg/max (68) - Timeout: 700 sec.
INFO exp: Progress(2/0/2): 100/100/100 min()/avg/max (68) - Timeout: 690 sec.
INFO exp: -----
INFO exp: Imaging Process Done
INFO exp: 2 nodes successfully imaged - Topology saved in '/tmp/pxe_slice-2022-01-21t23.25.51.359+00.0
INFO exp: -----
```

Figure 7: Image Loaded

After the image has been successfully loaded, you may power on all the required resources by:

```
yourusername@console:~$ omf tell -a on -t srv1-in1,srv1-in2,rfdev3-in1,rfdev3-in2,sdr2-in1,sdr2-in2
yourusername@console:~$ omf stat -t system:topo:allres
```

Then Figure 8 will be what you can see in the terminal.

```
Talking to the CMC service, please wait
-----
Node: rfdev3-in1.sb1.cosmos-lab.org      Reply: OK
Node: rfdev3-in2.sb1.cosmos-lab.org      Reply: OK
Node: sdr2-in1.sb1.cosmos-lab.org        Reply: OK
Node: sdr2-in2.sb1.cosmos-lab.org        Reply: OK
Node: srv1-in1.sb1.cosmos-lab.org        Reply: OK
Node: srv1-in2.sb1.cosmos-lab.org        Reply: OK
-----
```

Figure 8: Required Nodes Power On Successfully

And you will expect only the nodes that you powered on display as “POWERON” in the status list.

### 8.1.3 Connect the Switch

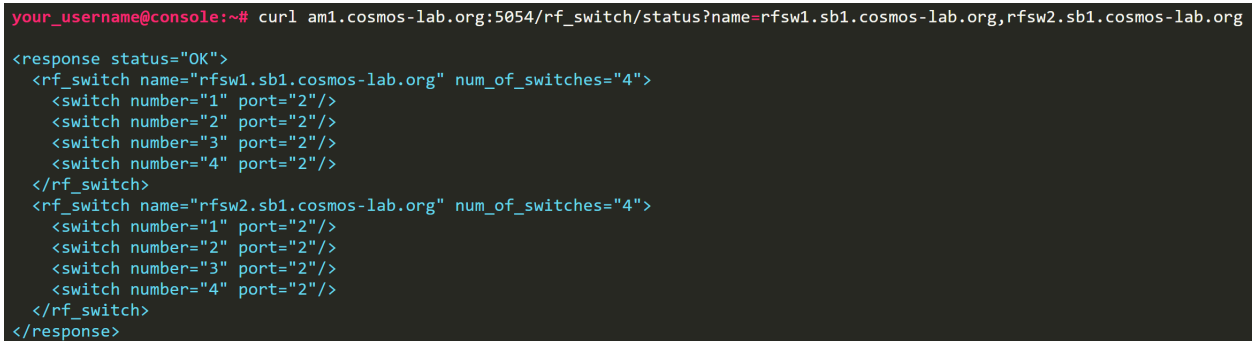
Since this tutorial is trying to use the RFSoc as the SDR, so we need to connect it to the RF front-end, which is the 60GHz Silvers array. According to Sec. 6.2, we use the following commands to connect the RFSoc with Silvers array (**Copy the command one line at a time, no space bar, and press enter after you paste all rows**):

```
curl "am1.cosmos-lab.org:5054/rf_switch/set?name=rfs1.sb1.cosmos-lab.org,
rfs2.sb1.cosmos-lab.org&switch=1,2,3,4&port=2"
```

You cannot check the switch status by using “omf stat” command. Instead, you should use the similar command with checking X-Y table status:

```
curl "am1.cosmos-lab.org:5054/rf_switch/status?name=
rfs1.sb1.cosmos-lab.org,rfs2.sb1.cosmos-lab.org"
```

What you want to see is what Figure 9 shows.



```
your_username@console:~# curl am1.cosmos-lab.org:5054/rf_switch/status?name=rfs1.sb1.cosmos-lab.org,rfs2.sb1.cosmos-lab.org
<response status="OK">
  <rf_switch name="rfs1.sb1.cosmos-lab.org" num_of_switches="4">
    <switch number="1" port="2"/>
    <switch number="2" port="2"/>
    <switch number="3" port="2"/>
    <switch number="4" port="2"/>
  </rf_switch>
  <rf_switch name="rfs2.sb1.cosmos-lab.org" num_of_switches="4">
    <switch number="1" port="2"/>
    <switch number="2" port="2"/>
    <switch number="3" port="2"/>
    <switch number="4" port="2"/>
  </rf_switch>
</response>
```

Figure 9: RF Switch Been Configured by connecting RFSoc and Silver array

## 8.2 RFSoc Setup

Every time the RFSoc is power cycled, you need to download the firmware/image from the server by using the Xilinx XSCT tool over JTAG. This process may take around 5 to 10 min. Remember, download one image at a time. When you access the root@ host, it will ask you if you want to proceed (fingerprint recognition). Just input “yes” and keep entering the rest of two row of commands.

Download package (necessary after power cycle the SDR) and configure the network:

```
yourusername@console_1:~$ ssh -Y root@srv1-in1
yourusername@console_2:~$ ssh -Y root@srv1-in2
```

```

cd mmwsdr ; git pull

root@srv1-in1:~/mmwsdr# cd fpga/nonrt-ch1/jtag/sb1_sdr2_in1
root@srv1-in2:~/mmwsdr# cd fpga/nonrt-ch1/jtag/sb1_sdr2_in2

xsct download_firmware.tcl

root@srv1-in1:~/mm..._in1# cd ../../../../host/scripts ; ./ipconfig_tx.sh
root@srv1-in2:~/mm..._in2# cd ../../../../host/scripts ; ./ipconfig_rx.sh

```

**Note:** If the ssh to root@node connection cannot be built, wait longer since the node might be still preparing for boot up.

**Note:** Use “exit” to disconnect from the sub-server.

**Note:** When there is an error during xsct downloading, try to power cycle the SDR node.

**Note:** Use “ip addr show” to double check if the network has been configured successfully (compared with the following commented values).

### 8.3 Basic Demos

For the one-node measurement:

**Run Xming server.**

```

root@srv1-in1:~# cd ../demos/basic
python onenode.py --f 60.48e9 -n srv1-in1 -m tx --min -450 --max 450

root@srv1-in2:~# cd ../demos/basic
python onenode.py -f 60.48e9 -n srv1-in2 -m rx -g 1 -d 1_3 -S y

```

For the two-node measurement: Put these command into srv1-in1 and srv1-in2 and run the twonode.py. The srv1-in1 will only communicate with the array, the actual commands will be sent from srv1-in2:

```

root@srv1-in1:~# cd ../mmwsdr/array ; python ederarray.py -u SN0240
root@srv1-in2:~# cd ../demos/basic
root@srv1-in2:~/mmwsdr/host/demos/basic#
python twonode.py -n srv1-in2 -N 1024 -F 5 -T cosmos -I 3 -g 2 -d 6_1 -S y
--x1 0 --y1 0 --a1 0 --x2 1300 --y2 0 --a2 0

python twonode.py -n srv1-in2 -N 1024 -F 5 -T cosmos -I 3 -g 2 -d 6_2 -S y
--x1 0 --y1 1300 --a1 0 --x2 1300 --y2 0 --a2 0

```

```
python twonode.py -n srv1-in2 -N 1024 -F 5 -T cosmos -I 3 -g 2 -d 6_3 -S y
--x1 0 --y1 0 --a1 0 --x2 1300 --y2 1300 --a2 0
```

```
python twonode.py -n srv1-in2 -N 1024 -F 5 -T cosmos -I 3 -g 2 -d 6_4 -S y
--x1 0 --y1 1300 --a1 0 --x2 1300 --y2 1300 --a2 0
```

## 8.4 Upload Data

After you run a measurement, you want to push the code to Github to update the data. First you use `ls` to check if the data is saved. Then use the following commands to update the Git.

## 8.5 Git

```
root@srv1-in2:~/mmwsdr/host/demos/basic# cd ../../..
root@srv1-in2:~/mmwsdr# git add .
root@srv1-in2:~/mmwsdr# git commit -m <message>
root@srv1-in2:~/mmwsdr# git push srv1 main
```

## 8.6 Save Images

Go to the server 1, use commands:

```
yourusername@console:~$ ssh root@srv1-in1
root@node:~# ./prepare.sh
root@node:~# exit
yourusername@console:~$ omf save -n srv1-in2.sb1.cosmos-lab.org
```

Then use WinSCP/command line to change the image name. The image is saved at: `/export/omf-images-5.4/`.

# 9 Tutorial: 60GHz USRP

For reference, check the original [tutorial](#).

## 9.1 Preparation

Before everything, put 4 terminals at the **laptop screen** and the **main monitor** respectively as the layout of Fig. 10 shows:



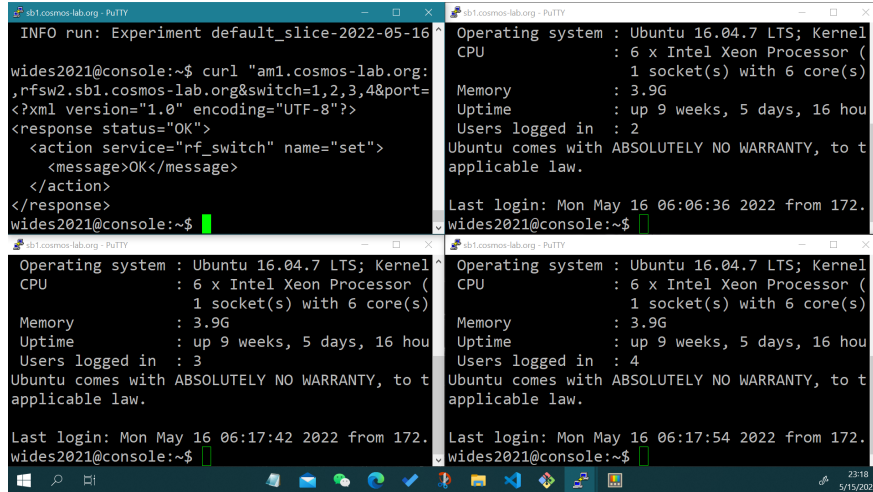


Figure 10: SSH terminals at laptop screen

For the laptop screen,

- upper left**            srv1-in1 monitor
- upper right**        srv1-in2 monitor
- bottom left**        srv1-in1 GUI starter
- bottom right**      srv1-in2 GUI starter

For the main monitor:

- upper left**            main controller
- upper right**        xytable movement controller
- bottom left**        srv1-in1 (Tx) controller
- bottom right**      srv1-in2 (Rx) controller

### 9.1.1 System Check and Reset

#### Big top left terminal

Turn off all devices and check status:

```
yourusername@console:~$ omf stat -t srv1-in1,srv1-in2,srv1-lg1,
srv2-lg1,sdr4-in1,sdr4-in2,rfdev3-in1,rfdev3-in2
```

```
yourusername@console:~$ omf tell -a offh -t srv1-in1,srv1-in2,srv1-lg1,  
srv2-lg1,sdr4-in2,rfdev3-in1,rfdev3-in2
```

### 9.1.2 Load Image and update Git

When the system is powered off, you can **load the system image into the needed servers** by using the command “omf load” (check your correct image from the root directory: /export/omf-images-5.4):

**Big upper left terminal**

```
yourusername@console:~$ omf load -i wides2021_USRP_lg1.ndz -t srv1-in1,srv1-in2
```

**Big bottom left terminal**

```
yourusername@console:~$ omf load -i wides2021_USRP_lg1.ndz -t srv1-lg1,srv2-lg1
```

Personal Git. Ettus UHD Git.

While the image is loading to the servers, update the Git from local to the remote by:

```
git status
```

If the Git is not updated, commit the current change and update the Git to the remote.

**If there is any change, commit and push them.**

### 9.1.3 Turn on Nodes

The 2 small servers should finish the image loading first, so after the image has been successfully loaded to srv1-in1 and srv1-in2, you may power on all the required resources by:

**Big upper left terminal**

```
yourusername@console:~$ omf tell -a on -t srv1-in1,srv1-in2,sdr4-in2,rfdev3-in1,rfdev3-in2
```

**Big bottom left terminal**

When the image is loaded to srv1-lg1 and srv2-lg1, it could take a while until them to be off. After check they are off, turn them on and check the status:

```
yourusername@console:~$ omf stat -t srv1-lg1,srv2-lg1
```

```
yourusername@console:~$ omf tell -a on -t srv1-lg1,srv2-lg1
```

```
yourusername@console:~$ omf stat -t srv1-in1,srv1-in2,srv1-lg1,  
srv2-lg1,sdr4-in1,sdr4-in2,rfdev3-in1,rfdev3-in2
```

### 9.1.4 Connect the Switch

#### Big upper left terminal

Configure the switch to make sure the Sivers frontend is connected to the USRP by port 1:

```
curl "am1.cosmos-lab.org:5054/rf_switch/set?name=rfs1.sb1.cosmos-lab.org,
rfs2.sb1.cosmos-lab.org&switch=1,2,3,4&port=1"
```

You cannot check the switch status by using “omf stat” command. Instead, you should use the similar command with checking X-Y table status:

```
curl "am1.cosmos-lab.org:5054/rf_switch/status?name=
rfs1.sb1.cosmos-lab.org,rfs2.sb1.cosmos-lab.org"
```

## 9.2 USRP Setup

SSH to the root servers for all terminals.

#### All terminals except Big bottom terminals(lefts are srv1-in1, rights are srv1-in2)

```
yourusername@console_1:~$ ssh -Y -o "StrictHostKeyChecking no" root@srv1-in1
yourusername@console_2:~$ ssh -Y -o "StrictHostKeyChecking no" root@srv1-in2
image passwd:          WIDES2021SRV1
```

The small server should be turned on first, then:

#### Big top 2 terminals

```
root@srv1-in*:~$ git pull; cd ~/uhd/host/build; make; sudo make install; sudo ldconfig
```

#### Big bottom 2 terminals

```
yourusername@console_1:~$ ssh -Y -o "StrictHostKeyChecking no" root@srv1-lg1
yourusername@console_2:~$ ssh -Y -o "StrictHostKeyChecking no" root@srv2-lg1
image passwd:          WIDES2021SRV1
root@srv*-lg1:~$ git pull; cd ~/uhd/host/build; make; sudo make install; sudo ldconfig
```

If there is no “build” folder, go with the **commented** codes:

### 9.2.1 GUI and video preparation

Once ready to run experiment, first you need to get the Sivers TRX-BF01 EVK GUI software ready. Access the GUI software.

## Run Xming server

### Small two bottom terminals

```
cd ~/ederenv; ./start_mb1.sh -gui SN0240
```

```
cd ~/ederenv; ./start_mb1.sh -gui SN0243
```

Within the 0240 GUI, choose the **Tx** tab, and then click **Enable Tx** and **LO Leakage Cal**. Within the 0243 GUI, click **Enable Rx**.

While the Tx is calculating the LO leakage, control the XY table by the **Small two upper terminals**.

```
cd ~/table/host/demos/basic/; python video.py -n srv1-in1
```

```
cd ~/table/host/demos/basic/; python video.py -n srv1-in2
```

### 9.2.2 Run Measurement

At the **Big two upper terminals**, control the XY table positions.

```
cd ~/table/host/demos/basic/
```

Control codes:

```
python move.py -g no -n srv1-in1 -c 1
```

```
python move.py -g no -n srv1-in2 -c 1
```

or control from single node:

```
python move.py -g yes -c 1
```

To check the XY tables positions and status, use:

```
python pos.py
```

When the Tx LO leakage calculation is done, go the **Big two bottom terminals**

=====

Now configure the network:

```
root@srv1-lg1:~# ~/uhd/host/examples/Brian/configuration_tx.sh
```

```
root@srv2-lg1:~# ~/uhd/host/examples/Brian/configuration_rx.sh
```

For single node:

```
~/uhd/host/examples/Brian/configuration_txrx.sh
```

At the **root@srv1-lg1**,

```
cp -v ~/uhd/host/examples/Brian/Signal/* ~/uhd/host/build/
```

Default:

```
~/uhd/host/build/examples/Brian_tx_2
```

```
~/uhd/host/build/examples/Brian_tx_2 --rate 200e6 --freq 100e6 --
```

```
signal cosmos_-100MHz_to_100MHz_SR_200MS --max-freq 100e6
```

At the **root@srv2-lg1**,

```
cp -v ~/uhd/host/examples/Brian/Signal/* ~/uhd/host/examples/Brian/Results/
```

```
cd ~/uhd/host/examples/Brian/Results; ~/uhd/host/build/examples/Brian_rx --  
rate 200e6 --freq 100e6 --nsamps 10000 --file test_1_1
```

```
cd ~/uhd/host/examples/Brian/Results; ~/uhd/host/build/examples/Brian_rx --  
rate 200e6 --freq 100e6 --duration 10 --file test_1_1
```

For the **Brian\_txrx.cpp**, run the commands below at **root@srv1-lg1**:

```
cp -v ~/uhd/host/examples/Brian/Signal/* ~/uhd/host/examples/Brian/Results/
```

```
cd ~/uhd/host/examples/Brian/Results; ~/uhd/host/build/examples/Brian_txrx --
rx-nsamps 10000 --rx-file-group 7 --rx-file-N 8 --rx-file-testid 1
```

=====

After the measurement, go to the GUI software and disable both Tx and Rx, then close the GUI softwares.

If you need to save the image, follow the Sec. 9.6. And turn off all nodes:

```
exit
```

```
yourusername@console:~$ omf tell -a offh -t srv1-in1,srv1-in2,srv1-lg1,
srv2-lg1,sdr4-in2,rfdev3-in1,rfdev3-in2
```

## 9.3 Attention

If need to update the FPGA image version:

```
/usr/local/lib/uhd/uhd_images_downloader.py
```

```
/usr/local/bin/uhd_image_loader --args="type=x300,addr=10.38.14.2"
```

Then power cycle the USRP after the update.

### 9.3.1 USRP Identification

Now use “uhd\_find\_devices” to find all available devices.

```
root@srv1-in1:~# uhd_find_devices
[INFO] [UHD] linux; GNU C++ version 7.5.0; Boost_106501; UHD_3.15.0.0-release
-----
-- UHD Device 0
-----
Device Address:
  serial: 31B6FFA
  addr: 10.38.14.2
  fpga: XG
  name: sdr4-in2
  product: X310
  type: x300
-----
-- UHD Device 1
-----
Device Address:
  serial: 31B8F3F
  addr: 10.38.14.1
  fpga: XG
  name: sdr4-in1
  product: X310
  type: x300
```

Figure 11: UHD Find Devices

You can either use serial, addr, or name to identify the USRPs.

### 9.3.2 Daughterboards

There are 4 different antenna modes for Tx and Rx.

- **Antenna Mode A:** real signal from antenna RX/Tx A
- **Antenna Mode B:** real signal from antenna RX/Tx B
- **Antenna Mode AB:** complex signal using both antennas (IQ)
- **Antenna Mode BA:** complex signal using both antennas (QI)

And the BW for the real-mode is 250 MHz, for complex-mode is 500MHz. For the measurement, the example gives the antenna mode as **AB**.

### 9.3.3 Subdevice

When specify a subdevice, the format is composed of

`<motherboard slot name>:<daughterboard frontend name>`

Both the daughterboards have one frontend:0, and just choose what example gives: **A:AB** for Tx and **B:AB** for Rx. The “AB” here means the different daughterboards, not antenna mode. The topology map of the daughterboard is [here](#). To know the exact name of the USRP devices and the daughter boards, use the following command:

```
uhd_usrp_probe
```

## 9.4 Upload Data

After you run a measurement, you want to push the code to Github to update the data. First you use `ls` to check if the data is saved. Then use the following commands to update the Git.

## 9.5 Git

```
root@srv1-in2:cd ~; git add .
root@srv1-in2: git commit ...
root@srv1-in2: git push
```

## 9.6 Save Images

**Make sure the GUI software are closed!!!** Go to the server that you want to save the image, use commands:

```
yourusername@console:~$ ssh root@srv1-in*
root@node:~# ./prepare.sh
root@node:~# exit
yourusername@console:~$ omf save -n srv1-in2.sb1.cosmos-lab.org
yourusername@console:~$ cd /export/omf-images-5.4; ls
yourusername@console:/export/omf-images-5.4$ mv *** wides2021_usrp_newbuild.ndz
```

Then use WinSCP/command line to change the image name. The image is saved at: **/export/omf-images-5.4/**.

```
yourusername@console:~$ omf tell -a offh -t srv1-in1,srv1-in2,srv1-lg1,
srv2-lg1,sdr4-in2,rfdev3-in1,rfdev3-in2
```

# 10 Measurement Required Devices

## 10.1 XY table

For XY table, right now the only available RF front-end is the Sivers 60GHz array. The SDRs are available with the RFSoc and X310 USRP.







```

INFO Experiment: Resetting resources
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [0 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [10 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [20 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [30 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [40 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [50 sec.]
INFO stdlib: Waiting for nodes (Up/Down/Total): 0/2/2 - (still down: srv1-in1.sbl.cosmos-lab.org,srv1-in2.sbl.cosmos-lab.org) [60 sec.]
INFO ALL_UP: Event triggered. Starting the associated tasks.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 790 sec.
INFO BRING_UP: Event triggered. Starting the associated tasks.
INFO Experiment: Bringing up resources
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 780 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 770 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 760 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 750 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 740 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 730 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 720 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 710 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 700 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 690 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 680 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 670 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 660 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 650 sec.

```

(a) Time Out Error 1

```

INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: 0 sec.
INFO exp: Progress(0/0/2): 0/0/0 min(srv1-in1.sbl.cosmos-lab.org)/avg/max (69) - Timeout: -10 sec.
INFO exp: -----
INFO exp: Imaging Process Done
INFO exp: 2 nodes timed out - Topology saved in '/tmp/pxe_slice-2022-01-17t21.25.31.710+00.00-topo-timeout.rb'
INFO exp: -----
INFO EXPERIMENT_DONE: Event triggered. Starting the associated tasks.
INFO NodeHandler:
INFO NodeHandler: Shutting down experiment, please wait...
INFO NodeHandler:
INFO NodeHandler: Shutdown flag is set - Turning Off the resources
INFO run: Experiment pxe_slice-2022-01-17t21.25.31.710+00.00 finished after 14:53

```

(b) Time Our Error 2

Figure 12: Time Out Error

## 11 Error Records

### 11.1 Timeout when loading the image to servers

#### 11.1.1 Description

When run the command below, you may encounter a time-out error by the Figure 12 shows.

```
yourusername@console:~$ omf load -i rfsoc_sivers_sb1.ndz -t srv1-in1,srv1-in2
```

#### 11.1.2 Solution

There is no solution needed. The reason for this error is that the back-end services are restarting periodically. By Jakub Kolodziejcki from the Rutgers University, “*just try the command again in a few minutes and it should work*”.

However, it also might be an image-error. Try to contact COSMOS if this error lasts more than 2 days.

## 11.2 Node Not Available Error

Maintenance Issue. Check with COSMOS to fix.

## 11.3 “No route to host” Error

Optical Switch Issue. Check with COSMOS to fix.. Or wait for the server to boot up completely.

## References

- [1] COSMOS. Service: Xy table. [Online]. Available: <https://wiki.cosmos-lab.org/wiki/Resources/Services/XYTable>