Analyse Single Coplanar Plane Wave resonator (CPW) using pyEPR

We use QISKIT METAL along with pyEPR

EPR: Energy Participation Ratio: the ledger book of energy deposition that changes Hamiltonian

In [1]:
```
import pyEPR as epr
import qiskit_metal as metal
from qiskit_metal import designs, draw
from qiskit_metal import Dict, Headings, MetalGUI
from qiskit_metal.analyses.quantization import EPRanalysis
from qiskit_metal.qlibrary.qubits.transmon_pocket import TransmonPocket
from qiskit_metal.qlibrary.terminations.open_to_ground import OpenToGround
from qiskit_metal.qlibrary.tlines.meandered import RouteMeander
```

In [2]:
```
#### Design and Build one resonator
```

In [3]:
```
design = designs.DesignPlanar({}, True)
design.chips.main.size['size_x'] = '2mm'
design.chips.main.size['size_y'] = '2mm'
gui =MetalGUI(design)
```

Creating one Transmon

In [4]:
```
design.delete_all_components()
q1=TransmonPocket(design, 'Q1',
                  options=dict(pad_width='425 um', pocket_height='650 um',
                               connection_pads=dict(readout=dict(loc_w=+1, l
gui.rebuild()
gui.autoscale()
```

Creating one CPW resonator and connecting to the transmon

In [5]:
```
otg = OpenToGround(design, 'open_to_ground', options=dict(pos_x='1.75mm',  p
RouteMeander(design, 'readout',  Dict(
        total_length='6 mm',
        hfss_wire_bonds = True,
        fillet='90 um',
        lead = dict(start_straight='100um'),
        pin_inputs=Dict(
            start_pin=Dict(component='Q1', pin='readout'),
            end_pin=Dict(component='open_to_ground', pin='open')), ))

gui.rebuild()
gui.autoscale()
```

Creating one analysis object dedicated to the readout

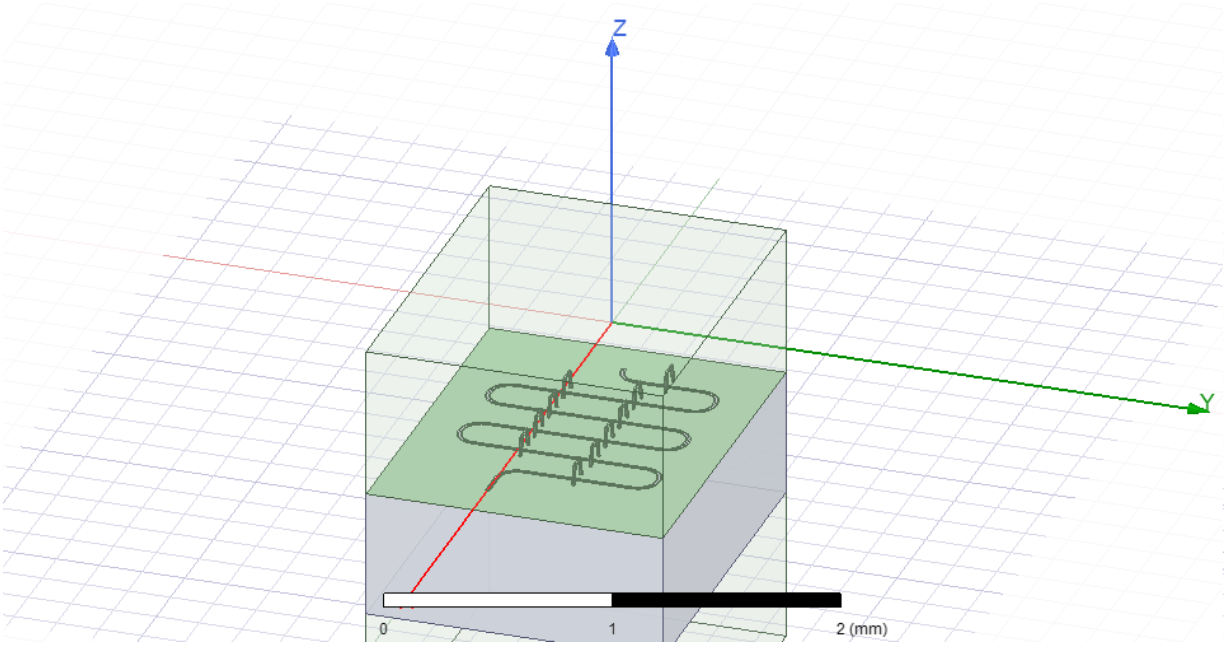
- Readout will beb analyzed in isolated condition
- Readout will be terminated with an "OPEN" on both the ends
- This analysis is being selected for both "readout" component and "open-to-ground" component.

In [6]:
```
eig_rd = EPRanalysis(design, "hfss")
```

In [7]:
```
eig_rd.sim.run(name="Readout",
               components=['readout', 'open_to_ground'],
               open_terminations=[('readout', 'start'), ('readout', 'end')])
eig_rd.sim.plot_convergences()
```

```
INFO 02:42PM [connect_project]: Connecting to Ansys Desktop API...
INFO 02:42PM [load_ansys_project]:      Opened Ansys App
INFO 02:42PM [load_ansys_project]:      Opened Ansys Desktop v2020.2.0
INFO 02:42PM [load_ansys_project]:      Opened Ansys Project
        Folder:    D:/Users/Abhijit Bhattacharyy/Documents/ANSYS_EM/
        Project:   Project1
INFO 02:42PM [connect_design]: No active design found (or error getting acti
ve design).
INFO 02:42PM [connect]:          Connected to project "Project1". No design
detected
INFO 02:42PM [connect_design]:  Opened active design
        Design:    Readout_hfss [Solution type: Eigenmode]
WARNING 02:42PM [connect_setup]:        No design setup detected.
WARNING 02:42PM [connect_setup]:        Creating eigenmode default setup.
INFO 02:42PM [get_setup]:       Opened setup `Setup`  (<class 'pyEPR.ansys.H
fssEMSetup'>)
INFO 02:42PM [get_setup]:       Opened setup `Setup`  (<class 'pyEPR.ansys.H
fssEMSetup'>)
INFO 02:42PM [analyze]: Analyzing setup Setup
02:43PM 41s INFO [get_f_convergence]: Saved convergences to D:\Qiskit_Metals
\TIFR\School\SingleResonator\hfss_eig_f_convergence.csv
```

In [8]:
```
eig_rd.sim.save_screenshot()
```



Out[8]: WindowsPath('D:/Qiskit_Metals/TIFR/School/SingleResonator/ansys.png')

Recover Eigenmode frequencies for each variations

In [9]:
```
eig_rd.get_frequencies()
```

```
Design "Readout_hfss" info:
        # eigenmodes    1
        # variations    1
Design "Readout_hfss" info:
        # eigenmodes    1
        # variations    1
```

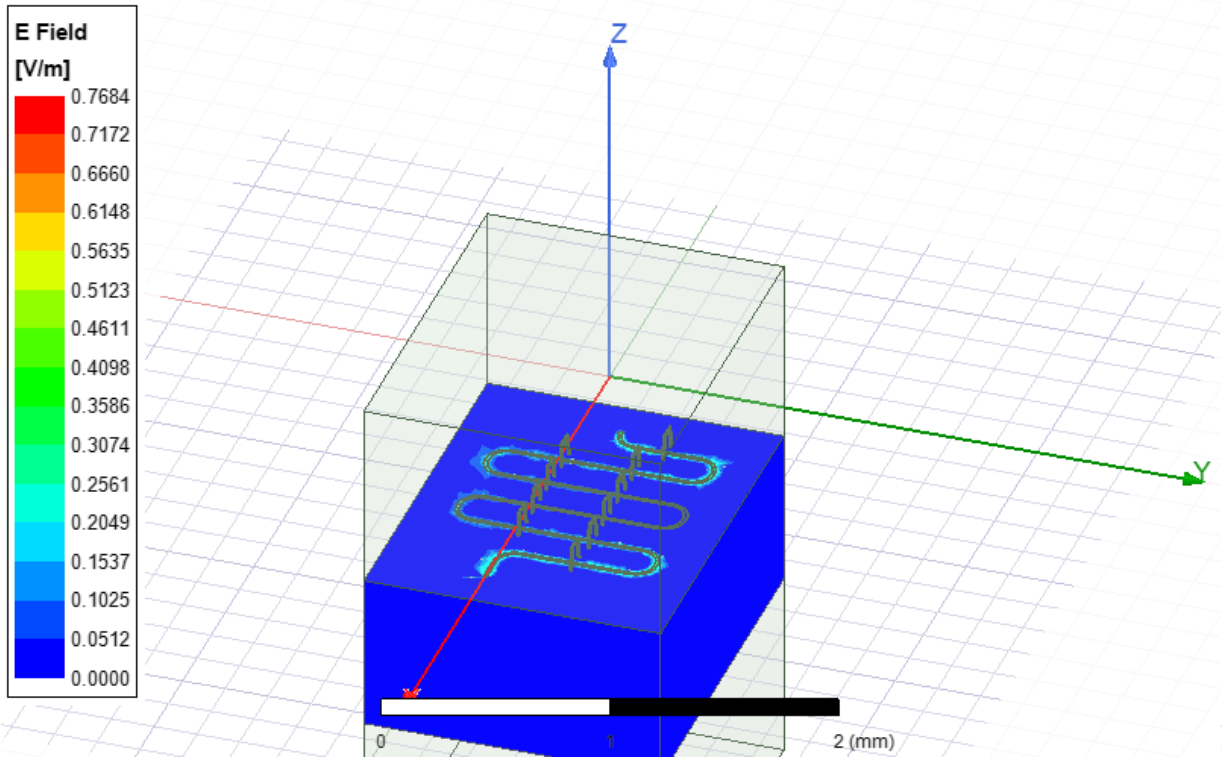
Out[9]:

		Freq. (GHz)	Quality Factor
variation	**mode**		
0	**0**	9.699888	inf

Plot E-Field on the Chip surface

In [10]:
```
eig_rd.sim.plot_fields('main')
eig_rd.sim.save_screenshot()
```

```
INFO 02:47PM [get_setup]:       Opened setup `Setup`  (<class 'pyEPR.ansys.H
fssEMSetup'>)
```



Out[10]: WindowsPath('D:/Qiskit_Metals/TIFR/School/SingleResonator/ansys.png')

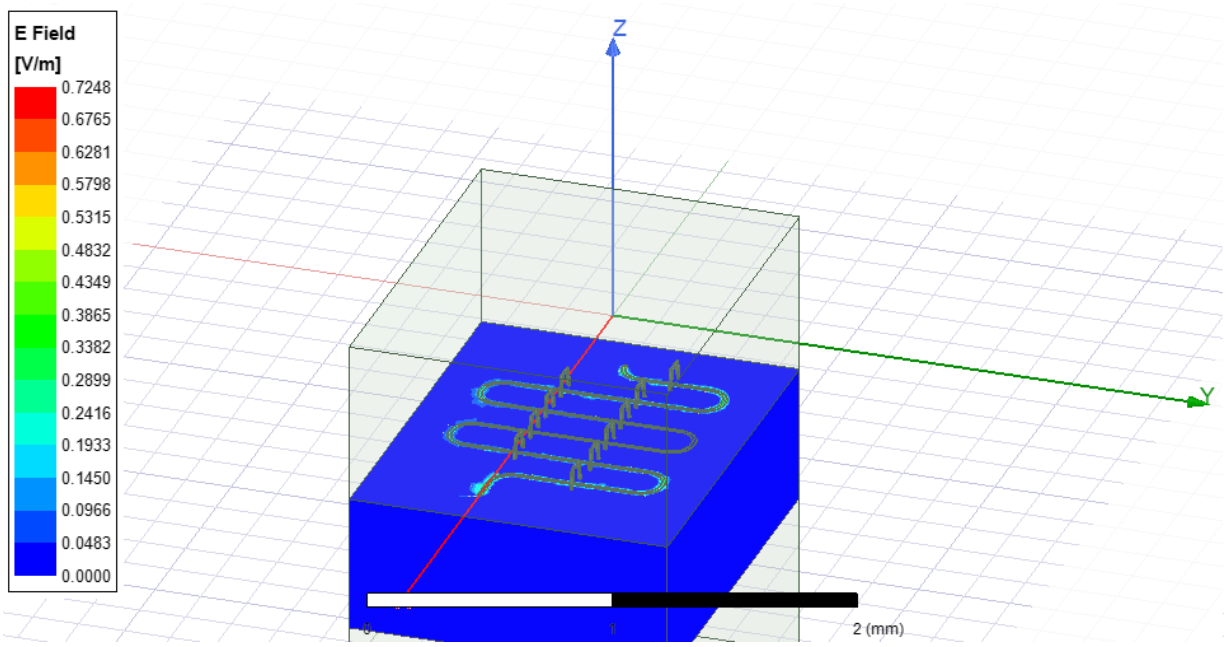
If EM field is not clear one may update number of passes and re-run

In [11]:
```
eig_rd.sim.setup.max_passes = 15
eig_rd.sim.run()
eig_rd.sim.plot_convergences()
```

```
INFO 02:49PM [get_setup]:       Opened setup `Setup`  (<class 'pyEPR.ansys.H
fssEMSetup'>)
INFO 02:49PM [analyze]: Analyzing setup Setup
02:51PM 17s INFO [get_f_convergence]: Saved convergences to D:\Qiskit_Metals
\TIFR\School\SingleResonator\hfss_eig_f_convergence.csv
```

In [12]:
```
eig_rd.sim.plot_fields('main')
eig_rd.sim.save_screenshot()
```

```
INFO 02:52PM [get_setup]:       Opened setup `Setup`  (<class 'pyEPR.ansys.H
fssEMSetup'>)
```



Out[12]: WindowsPath('D:/Qiskit_Metals/TIFR/School/SingleResonator/ansys.png')

EPR Analysis to find electric and Magnetic field energy stored in the readout system

In [13]:
```
eig_rd.run_epr(no_junctions=True)
```

```
Design "Readout_hfss" info:
        # eigenmodes    1
        # variations    1
Design "Readout_hfss" info:
        # eigenmodes    1
        # variations    1

        energy_elec_all       = 3.54608697113699e-24
        energy_elec_substrate = 3.23234658483469e-24
        EPR of substrate = 91.2%

        energy_mag    = 3.54608758919348e-24
        energy_mag % of energy_elec_all  = 100.0%
```

In [ ]:
```
```

In [15]:
```
gui.main_window.close()
eig_rd.sim.close()
```

In [ ]:
```
```