How to use Cad.py

Cad.py is an interface used to programmatically create photonic circuits. It can be used to create any kind of segment, specifying its size, index, offset and width.

To use the interface you have to import various classes:

from Cad import Circuit from Structs import *

Circuit, which is the interface.

• The file Structs.py contains classes used to define some properties of the photonic circuit

To run the code a specific Python interpreter must be used, which is contained in this directory:

How to run the code

C:/Synopsys/PhotonicSolutions/2019.09/RSoft/bin/rspython.exe Methods

Constructor

c1 = Circuit(dimension=d, dimension_y=n, in_file_name="index_file", out_file_name="out_file") Where with the dimension you can specify if the design is 2D or 3D (2 or 3), dimension_y is the height of the segments,

in_file_name is the name of the ".ind" file (RSoft format) without the file extension and the out_file_name is the name of the file with the simulation results. Add a segment

c1.add_segment(pos=vec3(x, y, z), offset=vec3(x, y, z), dimension_x=n, index=i)

where c1 is the circuit object previously created, pos=vec3(x, y, z) is the position of the segment, offset=vec3(x, y, z) is the

To add a segment to the circuit you can use the following method:

offset of the segment, dimension_x is the width of the segment and the index is the refractive index of the segment.

Add a pathway c1.add_pathway(segments)

Add a monitor to a pathway

Where segments is an array of strings with the segments you want in that pathway:

c1.add_pathway(["1", "4", "8"])

To add a monitor to a pathway you first need to create a pathway, then you can create the monitor: c1.add_monitor(pathway, monitor_type.TYPE, tilt, monitor_component.COMPONENT)

is the tilt of the monitor and monitor_component.COMPONENT is the component of the monitor (minor or major). Example: c1.add_monitor(1, monitor_type.MONITOR_FILE_POWER, "0", monitor_component.COMPONENT_MINOR)

You can find all of the "monitor_type "and monitor_component here.

Add a Lauch Field

• If you don't create a launch field by calling this function there the default one will stil be created, which has position (0, 0, 0) and a

Where pathway is the number of the pathway you want to add the monitor to, monitor_type.TYPE is the type of the monitor, tilt

• When you create the first lauch field, you just change the value of the default one.

c1.add_launch_field(x_position, power, launch_type.TYPE, launch_pathway, launch_tilt)

Run the Simulation

You can find all of the launch_type here.

To create a launch field you can use the following method:

you do not add a monitor it will return None

🦆 Main.py 🗙 🏺 Cad.py 1

🥏 Main.py > 🕅 main

When you create a launch field you have to pay attention to some things:

Example:

<u>∧ Attention!</u> ∴ This method returns an array which contains the last value all the monitors recorded, which means that if

c1.add_launch_field(0, 1, launch_type.LAUNCH_WGMODE, 0, 1)

c1.run_simulation() This method will run the simulation, create the .ind file and all of the files that contain simulation results.

from Cad import Circuit from Structs import *

c1 = Circuit(dimension=2, dimension_y=3, in_file_name="design", out_file_name="result")

c1.add_segment(pos=vec3(0, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(2, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2)

Once you have added all of the components to the circuit you can run the simulation with this method:

```
c1.add_segment(pos=vec3(4, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(6, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2)
                       c1.add_segment(pos=vec3(3, 0, 1), offset=vec3(0, 0, 7), dimension_x=7, index=2)
                        for i in [float(k) / 2 for k in range(4, 14, 1)]:
                            for j in[float(1) / 2 for 1 in range(1, 12, 1)]:
                                 c1.add_segment(pos=vec3(j, 0, i), offset=vec3(0, 0, 0.5), dimension_x=0.5, index=3)
                       c1.add_segment(pos=vec3(0, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)
                       c1.add_segment(pos=vec3(2, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(4, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(6, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)
                        for i in range(1,5):
                          c1.add_pathway([str(10*11+5+i)])
            25
                       c1.add_lauch_field(0, 1, launch_type.LAUNCH_WGMODE, 0, 1)
                       c1.add_lauch_field(2, 1, launch_type.LAUNCH_WGMODE, 0, 1)
c1.add_lauch_field(4, 1, launch_type.LAUNCH_WGMODE, 0, 1)
                        c1.add_lauch_field(6, 1, launch_type.LAUNCH_WGMODE, 0, 1)
                        print(c1.run_simulation())
            PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL
           PS C:\Users\RSOFT\Downloads\RSoft> & C:/Synopsys/PhotonicSolutions/2019.09/RSoft/bin/rspython.exe c:/Users/RSOFT/Downloads/RSoft/Main.py
           PS C:\Users\RSOFT\Downloads\RSoft>
Output file directory
The files will be created in a specific location. Once the simulation starts the /res directory will be generated in the folder where the .py
file is. Inside of it you will find other two folders.
       File Edit Selection
              EXPLORER
                         回の切り

✓ RSOFT

              > 🔯 .vscode
```

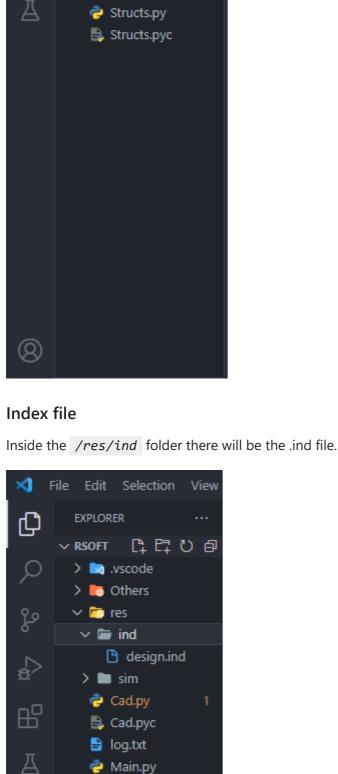
Cad.py Cad.pyc

مع

> 🌅 Others 🗸 🛅 res

> > ind > 🖿 sim

留 log.txt Main.py



Structs.py Structs.pyc

Results files Inside the /res/sim folder there will be all of the files genereted by running the simulation. The .mon file contains the data all the monitors captured. If there are no monitors in the circuit that file will not be present. File Edit Selection **EXPLORER** 中の世世 ✓ RSOFT > 🔯 .vscode > lo Others > ind 🗸 🔚 sim 0x result_xz.dat result_xz.pcs result_xz.pct result_xz.pdo result_xz.pdr result_xz.psf result_xz.psl result.dataindex result.mon result.pmn result.pmr 🥏 Cad.py

Cad.pyc log.txt 🥏 Main.py 🥏 Structs.py Structs.pyc

Special Parameters

 MONITOR_WGMODE_PHASE MONITOR_GAUSS_POWER MONITOR_GAUSS_PHASE MONITOR_LAUNCH_POWER MONITOR_LAUNCH_PHASE MONITOR_WG_POWER MONITOR_TOTAL_POWER MONITOR_FIELD_NEFF • MONITOR_FIELD_WIDTH MONITOR_FIELD_HEIGHT

Monitor Types and Monitor Components

MONITOR_FILE_POWER MONITOR_FILE_PHASE MONITOR_WGMODE_POWER

All the monitor types:

 COMPONENT_MINOR COMPONENT_MAJOR

All the monitor components:

Launch Types

All the launch types:

• LAUNCH_FILE

An example

 LAUNCH_COMPMODE • LAUNCH_WGMODE • LAUNCH_GAUSSIAN LAUNCH_RECTANGLE LAUNCH_MULTIMODE LAUNCH_PLANEWAVE

from Structs import * def main(): c1 = Circuit(dimension=2, dimension_y=3, in_file_name="design", out_file_name="result")

from Cad import Circuit

for j in[float(1) / 2 for 1 in range(1, 12, 1)]: c1.add_segment(pos=vec3(j, 0, i), offset=vec3(0, 0, 0.5), dimension_x=0.5, index=3) c1.add segment(pos=vec3(0, 0, 8), offset=vec3(0, 0, 1), dimension x=1, index=2) c1.add_segment(pos=vec3(2, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)

```
c1.add_segment(pos=vec3(6, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)
for i in range(1,5):
    c1.add_pathway([str(10*11+5+i)])
```

```
RSOFT
 > 🔯 .vscode
```

```
c1.add_lauch_field(0, 1, launch_type.LAUNCH_WGMODE, 0, 1)
        c1.add_lauch_field(2, 1, launch_type.LAUNCH_WGMODE, 0, 1)
        c1.add_lauch_field(4, 1, launch_type.LAUNCH_WGMODE, 0, 1)
        c1.add lauch field(6, 1, launch type.LAUNCH WGMODE, 0, 1)
        print(c1.run_simulation())
 main()
Result
This is the result of the code above:
                                    P Main.py X P Cad.py 1
                                     🥏 Main.py > 🕅 main
                                            from Cad import Circuit
                                             from Structs import launch_field_all, vec3, vec2, monitor_type, monitor_component, launch_type
         > DO Others
         🗸 🛅 res
           > ind
                                                 c1 = Circuit(dimension=2, dimension_y=3, in_file_name="design", out_file_name="result")
          ∨ 🗁 sim
              0x result_xz.dat
                                                 c1.add_segment(pos=vec3(0, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2)
                                                 c1.add_segment(pos=vec3(2, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(4, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(6, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2)
              result_xz.pcs
              result_xz.pct
              result_xz.pdo
               result_xz.pdr
                                                 c1.add_segment(pos=vec3(3, 0, 1), offset=vec3(0, 0, 7), dimension_x=7, index=2)
               result_xz.psf
              result_xz.psl
                                                  for i in [float(k) / 2 for k in range(4, 14, 1)]:
                                                       for j in[float(1) / 2 for 1 in range(1, 12, 1)]:
              result.dataindex
                                                           c1.add_segment(pos=vec3(j, 0, i), offset=vec3(0, 0, 0.5), dimension_x=0.5, index=3)
              result.mon
                                                  c1.add_segment(pos=vec3(0, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)
              result.pmr
                                                 c1.add_segment(pos=vec3(2, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(4, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(6, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)
            P Cad.py
            🗎 Cad.pyc
            log.txt
                                                  for i in range(1,5):
            Main.py
                                                      c1.add_pathway([str(10*11+5+i)])
            Structs.py
            Structs.pyc
                                                  c1.add_monitor(1, monitor_type.MONITOR_WGMODE_POWER, "1", monitor_component.COMPONENT_MINOR)
                                                 c1.add_monitor(2, monitor_type.MONITOR_WGMODE_POWER, "2", monitor_component.COMPONENT_MINOR)
c1.add_monitor(3, monitor_type.MONITOR_WGMODE_POWER, "3", monitor_component.COMPONENT_MINOR)
c1.add_monitor(4, monitor_type.MONITOR_WGMODE_POWER, "4", monitor_component.COMPONENT_MINOR)
                                     PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL
                                     PS C:\Users\RSOFT\Downloads\RSoft> & C:/Synopsys/PhotonicSolutions/2019.09/RSoft/bin/rspython.exe c:/Users/RSOFT/Downloads/RSoft/Main.py ['3.923551E-002', '2.366547E-002', '2.366547E-002', '3.923551E-002']
```

c1.add_segment(pos=vec3(0, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(2, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(4, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2) c1.add_segment(pos=vec3(6, 0, 0), offset=vec3(0, 0, 1), dimension_x=1, index=2)

c1.add_segment(pos=vec3(3, 0, 1), offset=vec3(0, 0, 7), dimension_x=7, index=2)

c1.add_segment(pos=vec3(4, 0, 8), offset=vec3(0, 0, 1), dimension_x=1, index=2)

for i in [float(k) / 2 for k in range(4, 14, 1)]: