# Laboratório 08: Acoplador direcional

Professor: Adolfo Fernandes Herbster

Aluno: Edilberto Elias Xavier Junior

Matrícula: 120210134

## 1. Objetivo

- gerar os componentes e objetos de simulação por meio da API Python;
- utilizar o Lumerical FDE para determinar o comprimento de acoplamento do acoplador direcional;
- utilizar o Lumerical FDTD para:

  – obter os desempenho final do dispositivo;

  – comparar os resultados gerados com aqueles gerados pelo varFDTD e;

  – gerar seus parâmetros S;
- utilizar o Lumerical INTERCONNECT para simular o dispositivo utilizando os parâmetros S gerados no FDTD..

## 2. Atividades

- **Modelo de simulação**

  Create Acoplador Direcional

```python
sub_material = 'SiO2 (Glass) - Palik'

gap = 200*nm

x_span_waveguide = 25*um
y_span_waveguide = 0.450*um
z_span_waveguide = 0.22*um
x_waveguide = 0.0*um
y_waveguide = ((gap+y_span_waveguide)/2)
z_waveguide = 0.0*um
waveguide_material = 'Si (Silicon) - Palik'
radiu = 10*um
```

Python

```python
modeApi.switchtolayout()

modeApi.deleteall()

modeApi.addrect()
modeApi.set('name', 'waveguide_topper')
modeApi.set('material', waveguide_material)
modeApi.set('x', x_waveguide)
modeApi.set('y', -y_waveguide)
modeApi.set('z', z_waveguide)
modeApi.set('x span', x_span_waveguide)
modeApi.set('y span', y_span_waveguide)
modeApi.set('z span', z_span_waveguide)

modeApi.copy()
modeApi.set('name', 'waveguide_upper')
modeApi.set('y', y_waveguide)
modeApi.set('x span', (x_span_waveguide-2*radiu))

modeApi.addobject('90_bend_wg')
modeApi.set('name', '90_bend_left')
modeApi.set('material', waveguide_material)
modeApi.set('x', -(x_span_waveguide-2*radiu)/2)
modeApi.set('y', (radiu+y_waveguide))
modeApi.set('z', z_waveguide)
modeApi.set('radius', radiu)
modeApi.set('base width', y_span_waveguide)
modeApi.set('base height', z_span_waveguide)
modeApi.set("first axis", 'z')
modeApi.set('rotation 1', 90)
modeApi.set('second axis', 'x')
modeApi.set('rotation 1',180)

modeApi.copy()
modeApi.set('name', '90_bend_rigth')
modeApi.set('x', (x_span_waveguide-2*radiu)/2)
modeApi.set('y', (radiu+y_waveguide))
modeApi.set("first axis", 'z')
modeApi.set('rotation 1', -90)
```

Python

Solver FDE

```python
fde_solver_material = sub_material

x_fde = 0.0*um
y_fde = 0.0*um
z_fde = 0.0*um
x_span_fde = 0*um
y_span_fde = 2.5*um
z_span_fde = 1*um

mesh_cells = 100
mesh_multiplier = 5

wavelength = 1550*nm
start_wavelength = 1550*nm
stop_wavelength = 1550*nm

modes = 4
```

Python

```python
modeApi.switchtolayout()

modeApi.select('FDE')
modeApi.delete()

modeApi.addfde()
modeApi.set('solver type', '2D X normal')
modeApi.set('background material', fde_solver_material)

modeApi.set('x', x_fde)
modeApi.set('y', y_fde)
modeApi.set('z', z_fde)
modeApi.set('z span', z_span_fde)
modeApi.set('y span', y_span_fde)

modeApi.set('define z mesh by', 'number of mesh cells')
modeApi.set('mesh cells z', mesh_cells)
modeApi.set('define y mesh by', 'number of mesh cells')
modeApi.set('mesh cells y', mesh_cells)

modeApi.set('wavelength', wavelength)

modeApi.set('number of trial modes', modes)

modeApi.set('fit materials with multi-coefficient model', True)
modeApi.set('wavelength start', start_wavelength)
modeApi.set('wavelength stop', stop_wavelength)

modeApi.set('z min bc', 'PML')
modeApi.set('z max bc', 'PML')
modeApi.set('y min bc', 'PML')
modeApi.set('y max bc', 'PML')
```

Python

```python
modeApi.switchtolayout()

modeApi.select('mesh')
modeApi.delete()

modeApi.addmesh()
modeApi.set('set mesh multiplier', True)
modeApi.set('x', x_fde)
modeApi.set('y', y_fde)
modeApi.set('z', z_fde)
modeApi.set('x span', x_span_fde)
modeApi.set('y span', y_span_fde)
modeApi.set('z span', z_span_fde)
modeApi.set('x mesh multiplier', mesh_multiplier)
modeApi.set('y mesh multiplier', mesh_multiplier)
modeApi.set('z mesh multiplier', mesh_multiplier)
```

Python

Solver FDTD

```python
fdtd_solver_material = sub_material

x_fdtd = 0.0*um
y_fdtd = 5.0*um
z_fdtd = 0.0*um
x_span_fdtd = (x_span_waveguide_acopla+2*radiu) + 2*um
y_span_fdtd = 1.2*radiu
z_span_fdtd = 5*um

mesh_accuracy = 2

start_wavelength = 1500*nm
stop_wavelength = 1600*nm
num_frequency_points = 21

time_simulation = (np.pi*radiu + x_span_waveguide_acopla + radiu*2 + 5*um)*7/c + 200e-15
```

✓ 0.0s Python

```python
fdtdApi.switchtolayout()

fdtdApi.select('FDTD')
fdtdApi.delete()

fdtdApi.addfdtd()
fdtdApi.set('background material', fdtd_solver_material)

fdtdApi.set('x', x_fdtd)
fdtdApi.set('y', y_fdtd)
fdtdApi.set('z', z_fdtd)
fdtdApi.set('x span', x_span_fdtd)
fdtdApi.set('y span', y_span_fdtd)
fdtdApi.set('z span', z_span_fdtd)

fdtdApi.set('mesh accuracy', mesh_accuracy)
fdtdApi.set('simulation time', time_simulation)

fdtdApi.set('x min bc', 'PML')
fdtdApi.set('x max bc', 'PML')
fdtdApi.set('y min bc', 'PML')
fdtdApi.set('y max bc', 'PML')
fdtdApi.set('z min bc', 'Symmetric')
fdtdApi.set('z max bc', 'PML')

fdtdApi.set('global source wavelength start', start_wavelength)
fdtdApi.set('global source wavelength stop', stop_wavelength)
```

✓ 3.3s Python

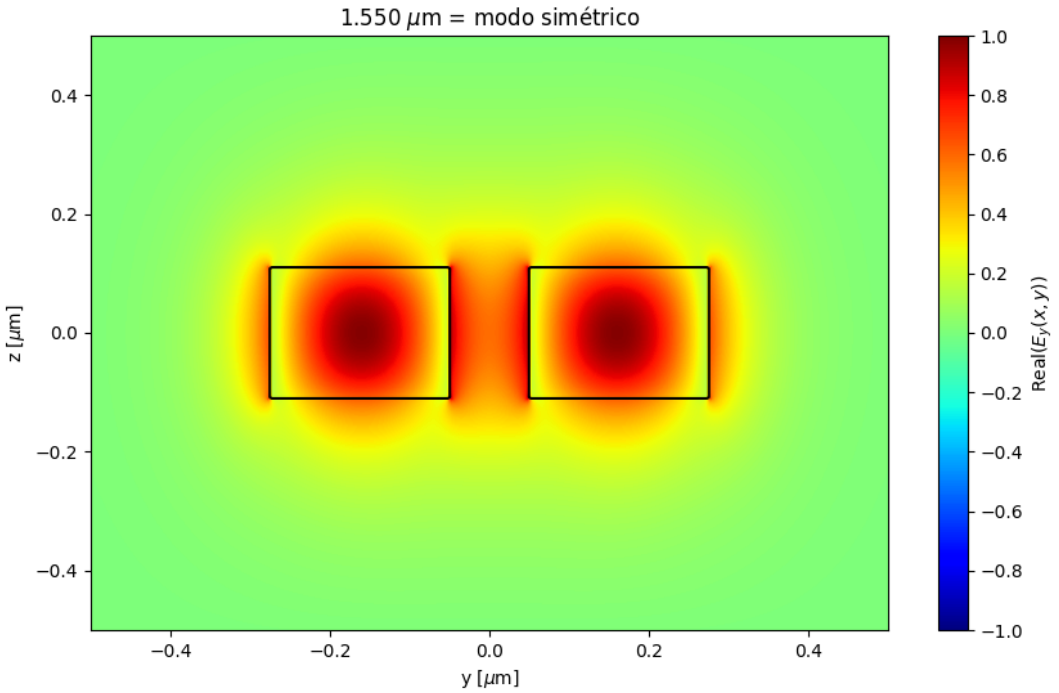
```python
fdtdApi.switchtolayout()

fdtdApi.select('Monitor_E')
fdtdApi.delete()

fdtdApi.addprofile()
fdtdApi.set('name', 'Monitor_E')
fdtdApi.set('override global monitor settings', True)
fdtdApi.set('frequency points', num_frequency_points)
fdtdApi.set('x', x_fdtd)
fdtdApi.set('y', y_fdtd)
fdtdApi.set('x span', x_span_fdtd)
fdtdApi.set('y span', y_span_fdtd)
fdtdApi.set('z', z_fdtd)
```

✓ 2.1s Python

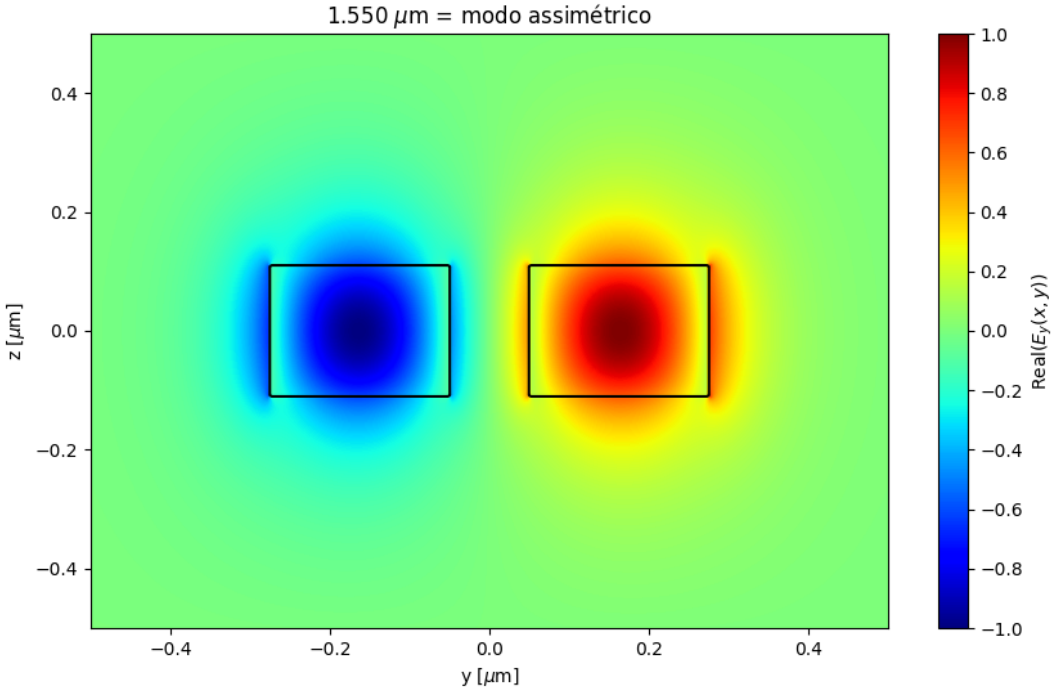
```
fdtdApi.switchtolayout()

fdtdApi.select('FDTD::ports')
fdtdApi.delete()


fdtdApi.addport()
fdtdApi.set('name', 'port_00')
fdtdApi.set('injection axis', 'y-axis')
fdtdApi.set('x', -(x_span_waveguide_acopla + 2*radiu)/2)
fdtdApi.set('x span', y_span_waveguide*2)
fdtdApi.set('y', (radiu + (y_span_waveguide + gap)/2))
fdtdApi.set('z', z_fdtd)
fdtdApi.set('z span', z_span_fdtd)
fdtdApi.set('direction', 'Backward')
fdtdApi.set('mode selection', 'fundamental TE mode')

fdtdApi.copy()
fdtdApi.set('name', 'port_01')
fdtdApi.set('x', (x_span_waveguide_acopla + 2*radiu)/2)

fdtdApi.addport()
fdtdApi.set('name', 'port_10')
fdtdApi.set('injection axis', 'x-axis')
fdtdApi.set('x', -(x_span_waveguide_acopla + 2*radiu)/2)
fdtdApi.set('y', -(y_span_waveguide + gap)/2)
fdtdApi.set('y span', y_span_waveguide*2)
fdtdApi.set('z', z_fdtd)
fdtdApi.set('z span', z_span_fdtd)
fdtdApi.set('mode selection', 'fundamental TE mode')
fdtdApi.set('direction', 'Forward')

fdtdApi.copy()
fdtdApi.set('name', 'port_11')
fdtdApi.set('x', (x_span_waveguide_acopla + 2*radiu)/2)
fdtdApi.set('direction', 'Backward')

fdtdApi.select('FDTD::ports')
fdtdApi.set('monitor frequency points', num_frequency_points)
```

✓ 4.4s Python

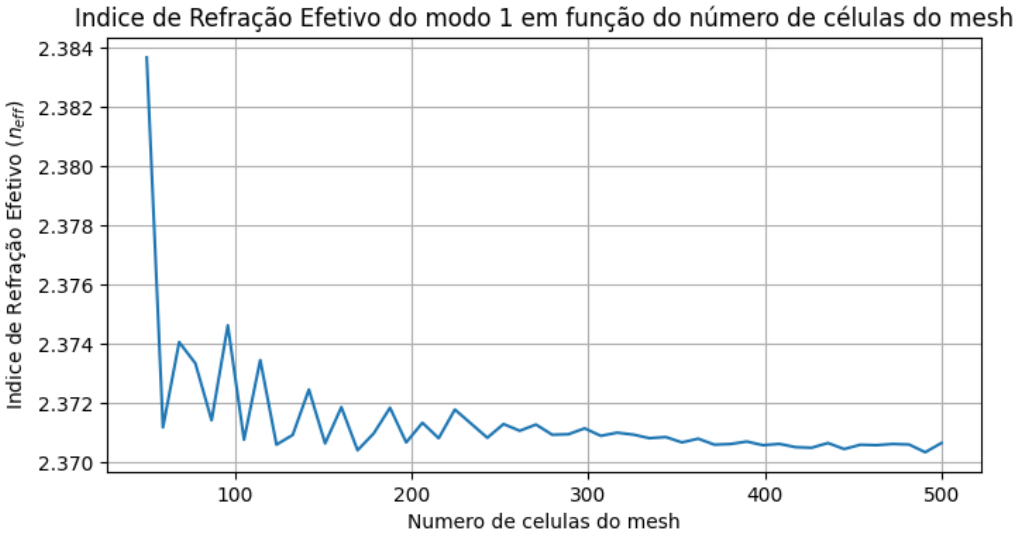
- **Simulação - solver FDE**
  **Simulação do dispositivo - características dos modos**



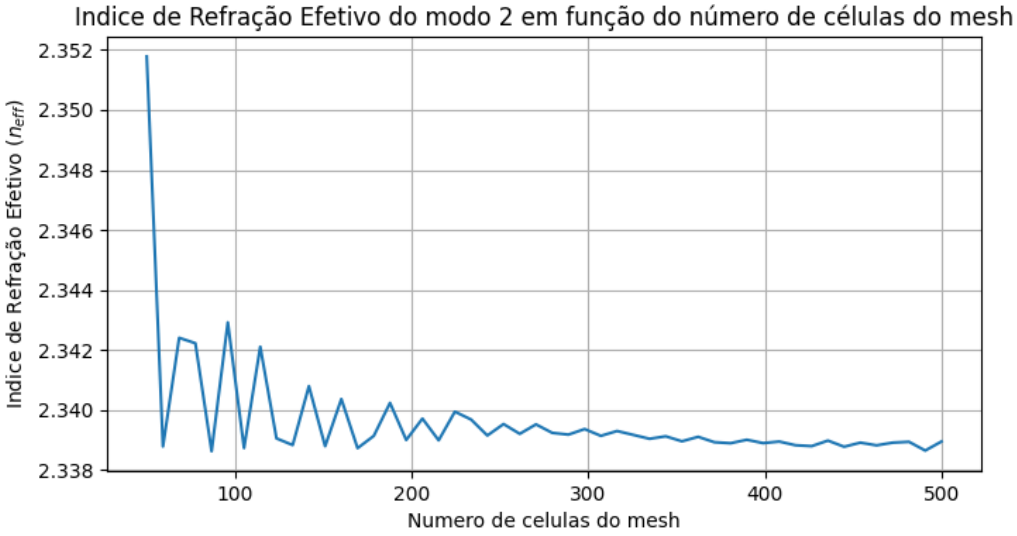

**Perfil de Campo Elétrico - FDE – MODO 1**




**Perfil de Campo Elétrico – FDE – MODO 2**

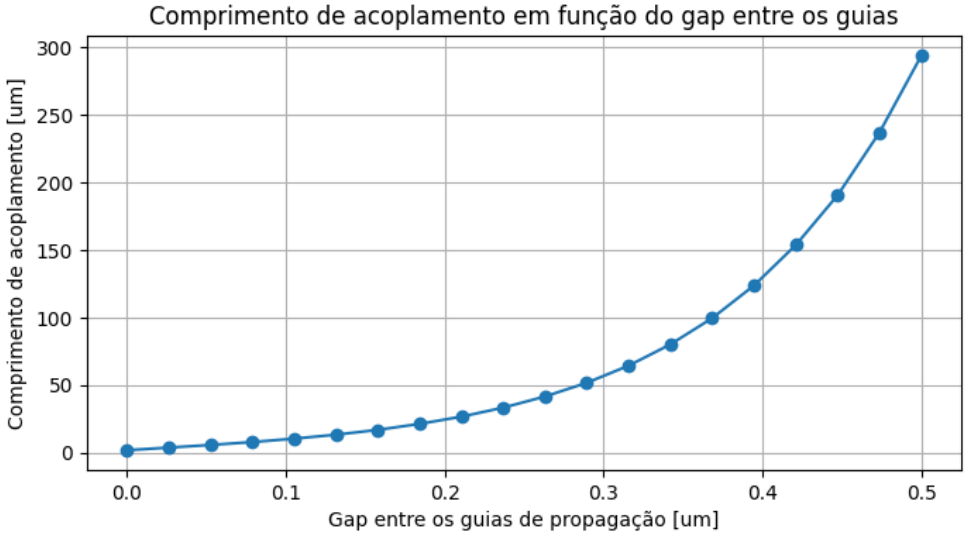



**Analise de Convergência do mesh – MODO 1**




**Analise de Convergência do mesh – MODO 2**

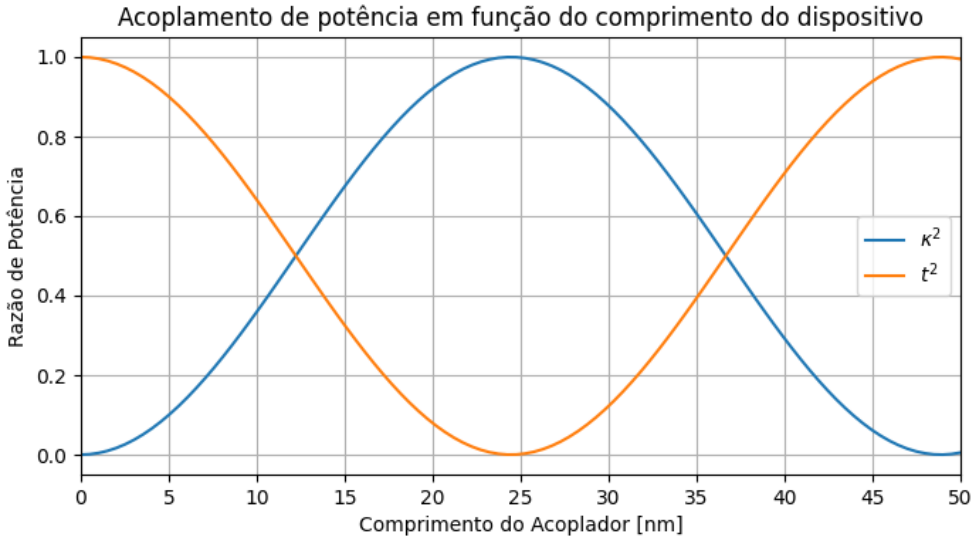
## Análise - comprimento de acoplamento em função da distância entre guias




**Comprimento do acoplador em função do gap entre os guias**

A partir do gráfico acima nota-se que quanto maior for a distância entre os guias (gap) maior terá que ser o comprimento do guia para o acoplador direcional.

**Análise - acoplamento de potência em função do comprimento do dispositivo**

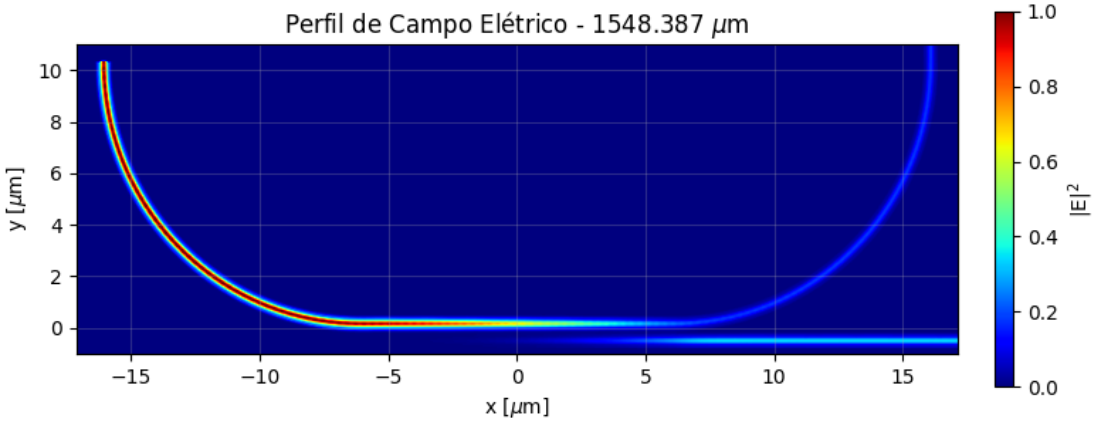



**Razão de Potência em Função do comprimento do acoplador**

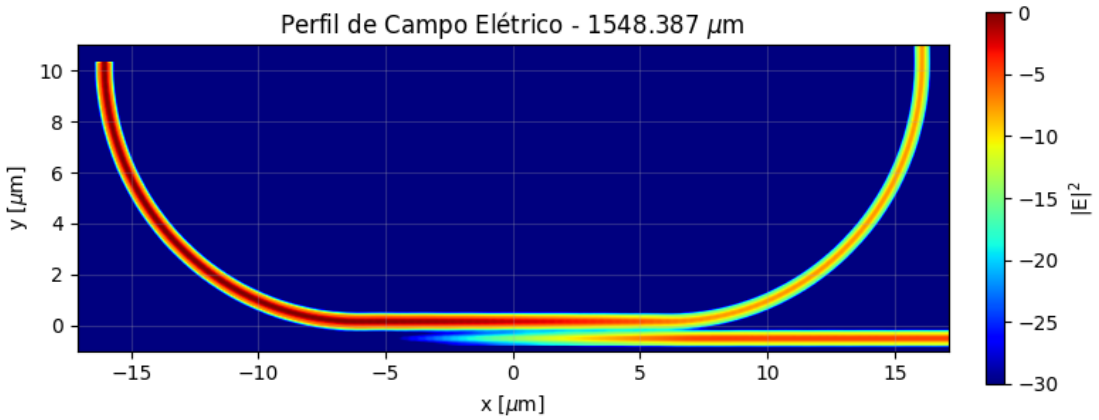
A partir do gráfico acima é possível determinar o menor comprimento para que se tenha o diversos tipos de razão de transferencia de pontencia entre os guias, como por exempo:

L_50/50 = 12.2208 µm – para uma razão 50/50
L_95/05 = 20.9326 µm - para uma razão 95/05
L_99/01 = 22.8830 µm - para uma razão 99/01

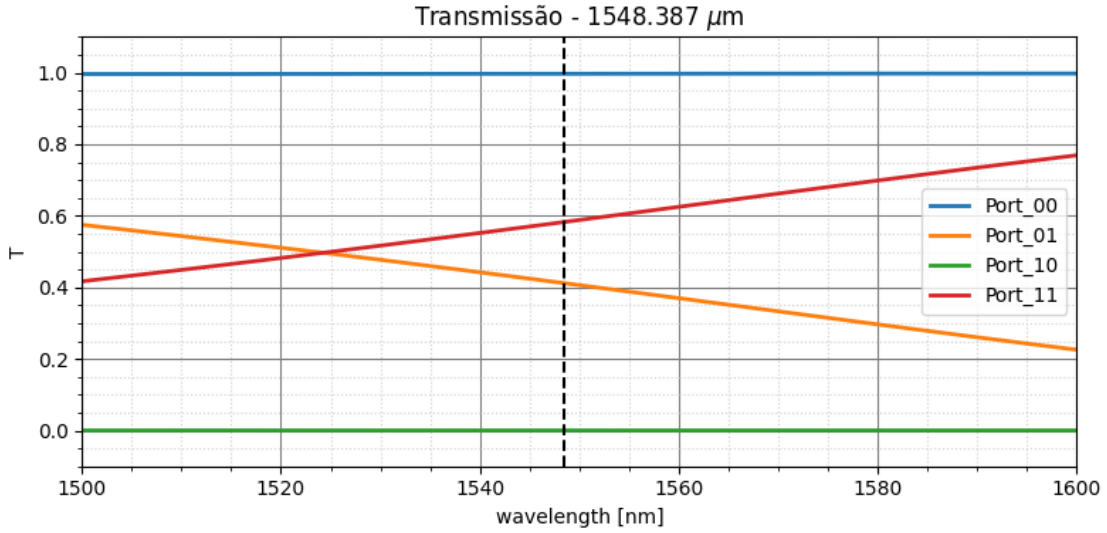
- **Simulação - solver FDTD**
  **Simulação do dispositivo – FDTD – MODO TE**



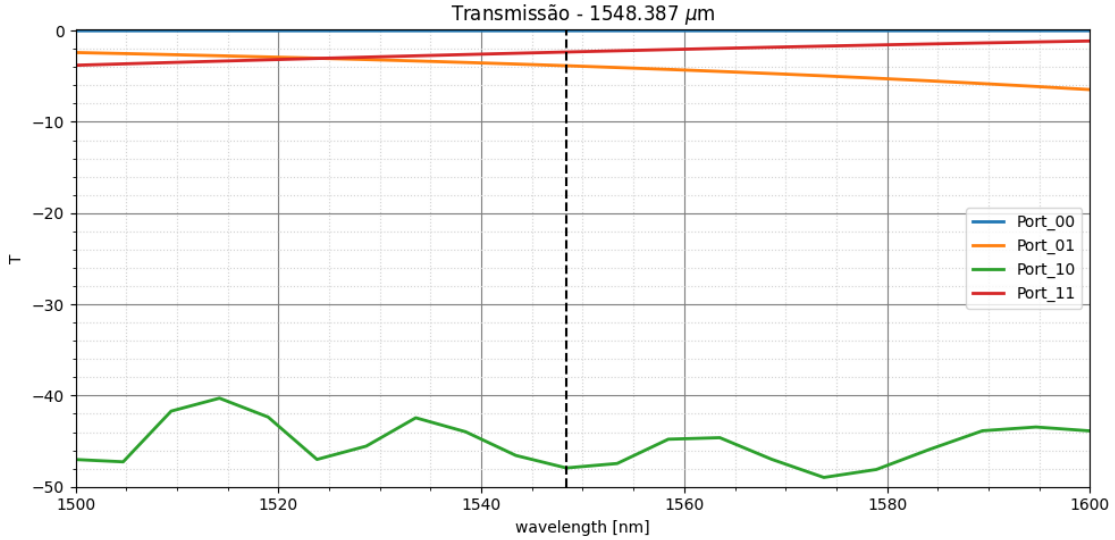

**Perfil de Campo Elétrico – FDTD – Linear – TE_MODE**




**Perfil de Campo Elétrico – FDTD – LOG – TE_MODE**




**Transmissão – Linear – FDTD – TE_MODE**




**Transmissão – LOG – FDTD – TE_MODE**

Aos 1550 nm, assim como em toda a faixa, a transmissão para cada porta é de:

Transmissão na Port_00 para 1550nm: 0.9969

Transmissão na Port_01 para 1550nm: 0.4061

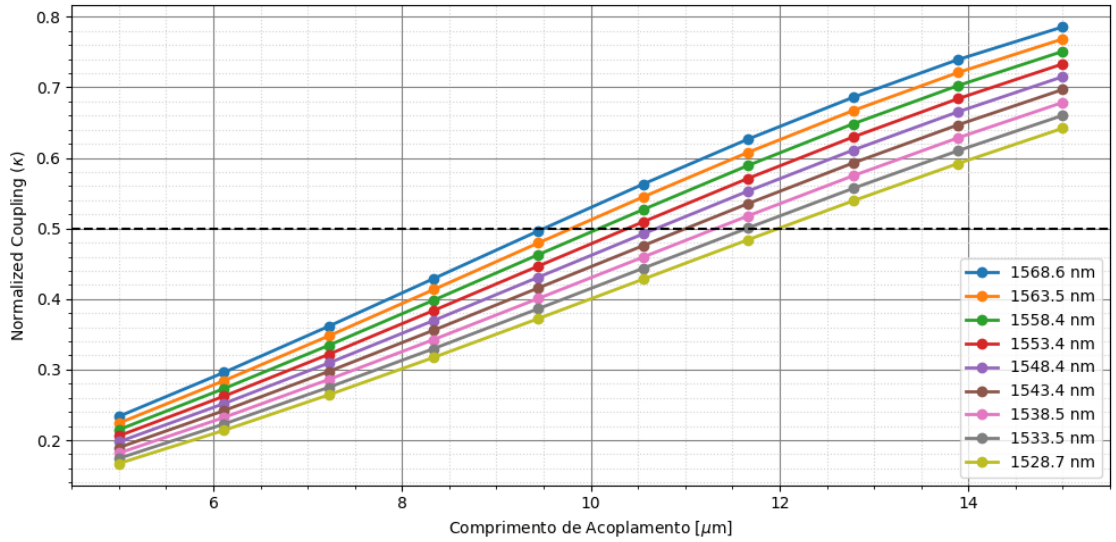
Transmissão na Port_10 para 1550nm: 0.0000

Transmissão na Port_11 para 1550nm: 0.5882

A perda por inserção, em dB, para o comprimento de onda de 1550 nm é de:

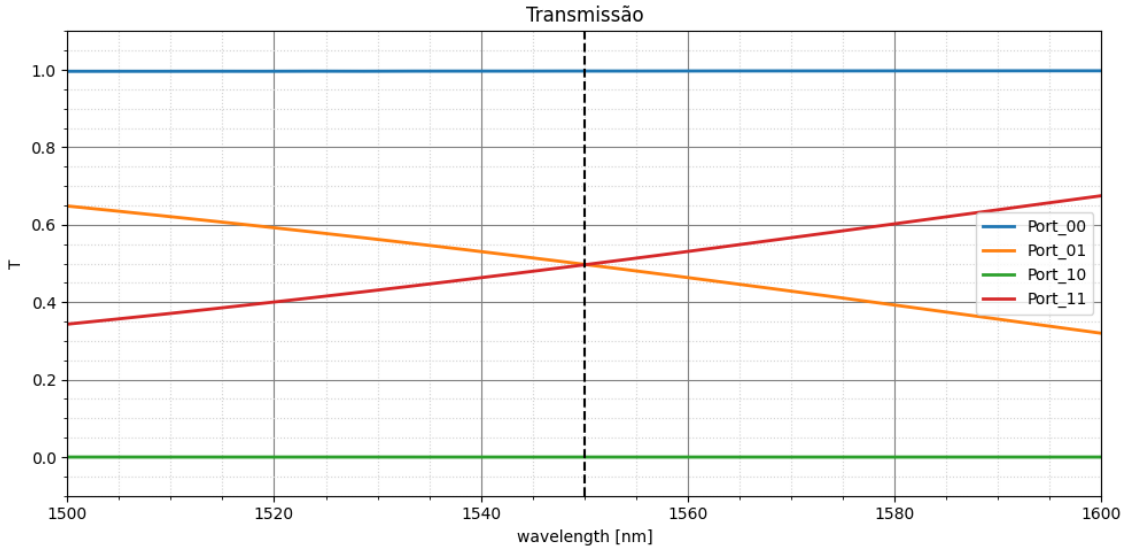
Perda de Inserção para 1550nm = -0.0248dB

## Otimização do dispositivo

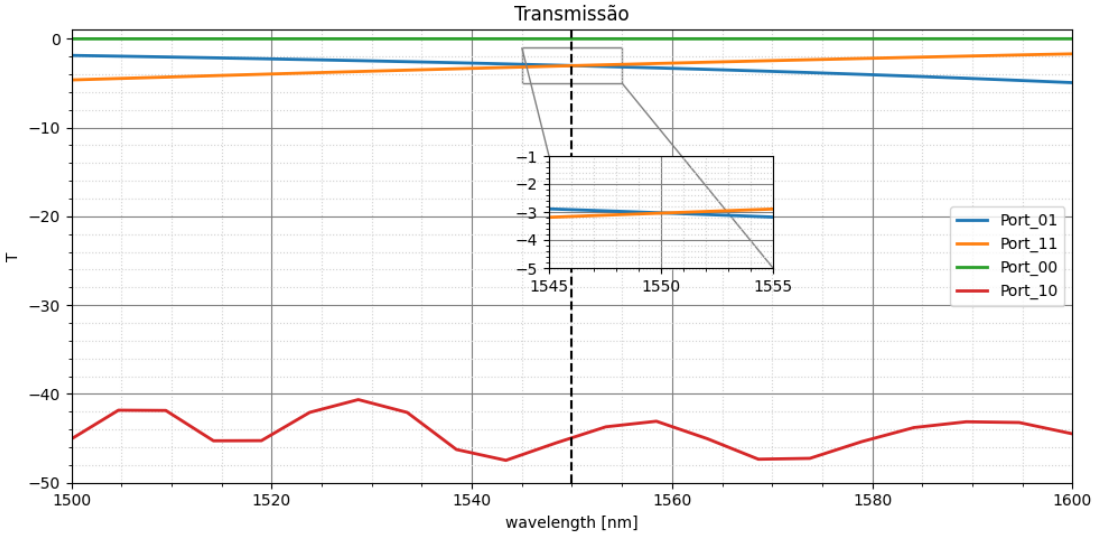



**Comprimento de Acoplamento para cada percentual de potencia tranferida – TE_MODE**

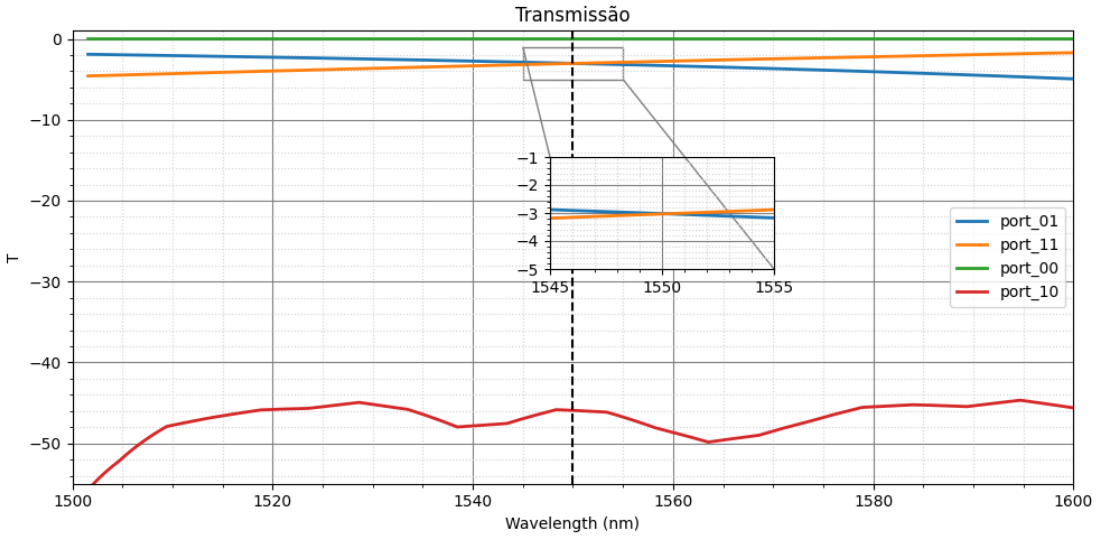
Como pode ser visto no gráfico acima o comprimento para uma razão 50/50 é de 10,53 microns




**Transmissão Otimizada – FDTD – Linear – TE_MODE**

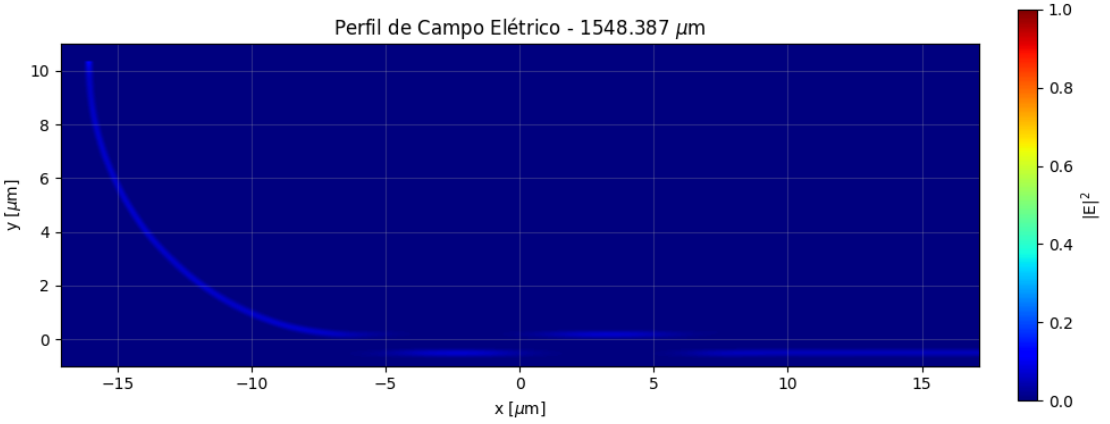



**Transmissão Otimizada – FDTD – Log – TE_MODE**

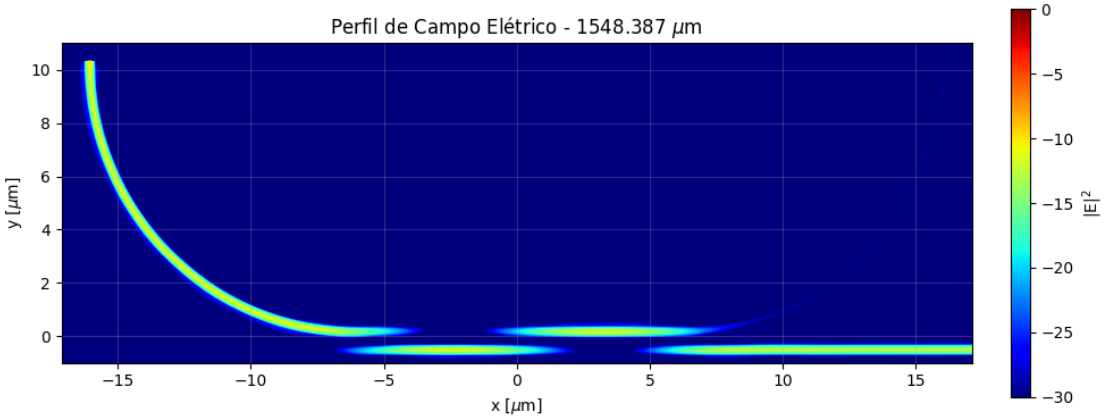



**Transmissão Otimizada – INTERCONCT – Log – TE_MODE**

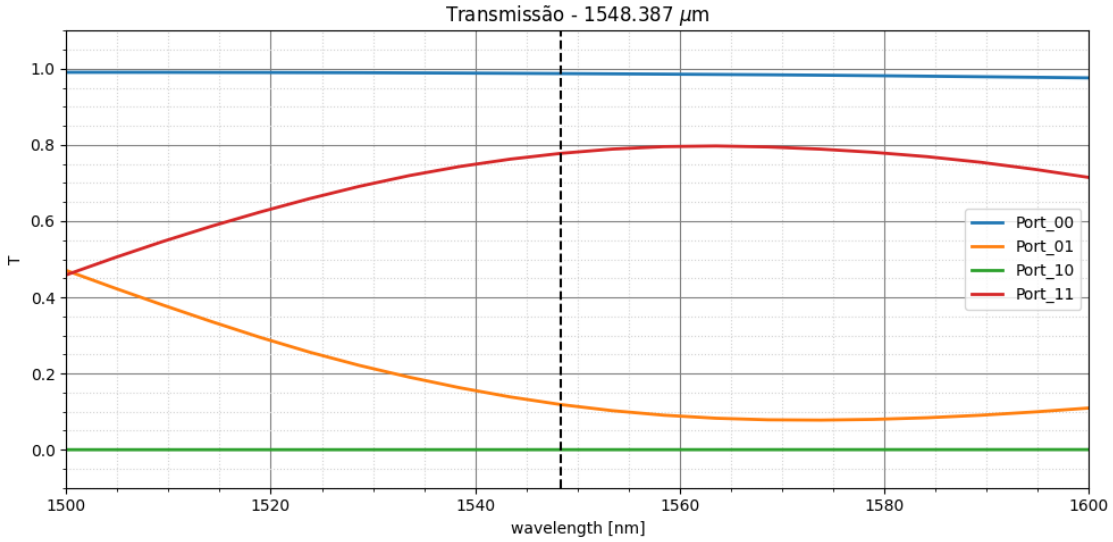
**Simulação do dispositivo – FDTD – MODO TM**



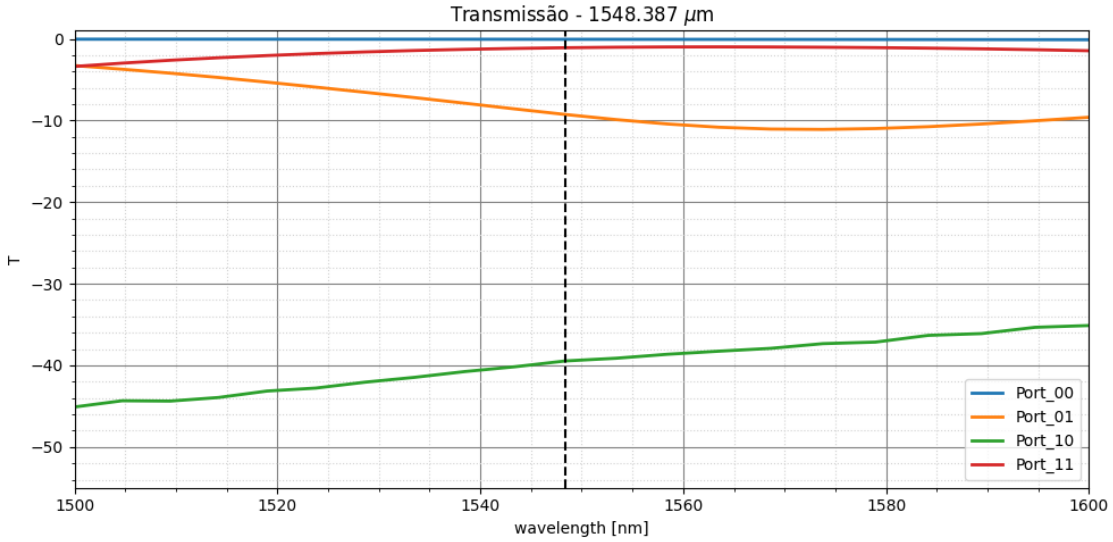

**Perfil de Campo Elétrico - FDTD - TM_MODE - Linear**




**Perfil de Campo Elétrico - FDTD - TM_MODE - LOG**




**Transmissão - Linear – FDTD – TM_MODE**




**Transmissão - LOG – FDTD – TM_MODE**

Aos 1550 nm, assim como em toda a faixa, a transmissão para cada porta é de:

Transmissão na Port_00 para 1550nm: 0.9872

Transmissão na Port_01 para 1550nm: 0.1133

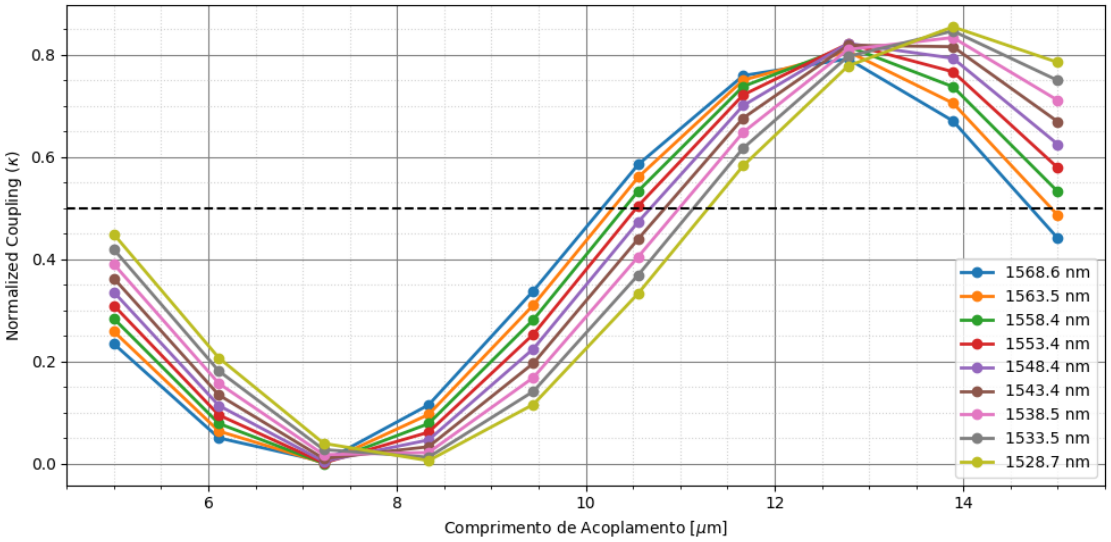
Transmissão na Port_10 para 1550nm: 0.0001

Transmissão na Port_11 para 1550nm: 0.7816

A perda por inserção, em dB, para o comprimento de onda de 1550 nm é de:

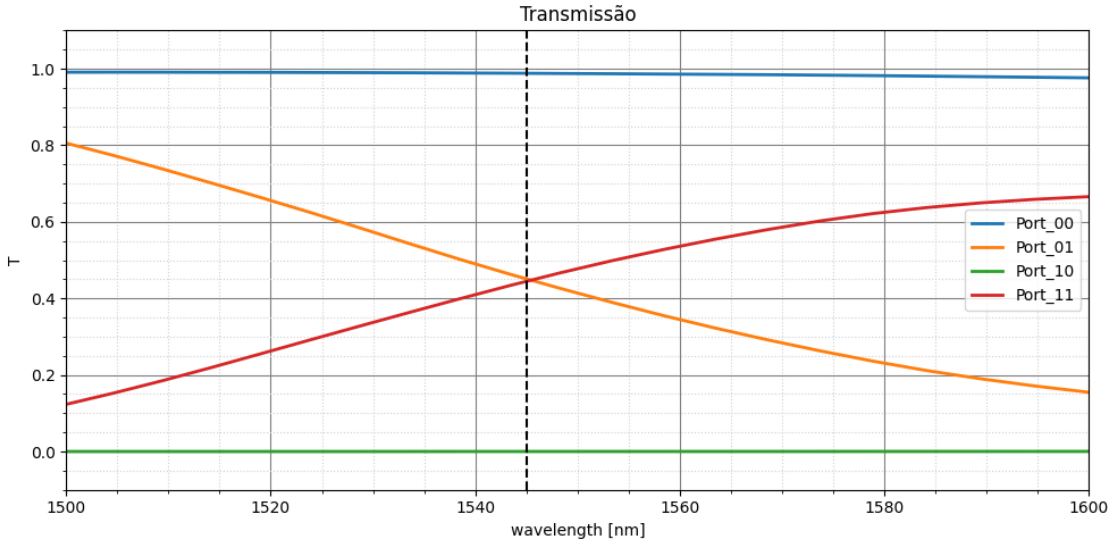
Perda de Inserção para 1550nm = -0.4824dB

## Otimização do dispositivo

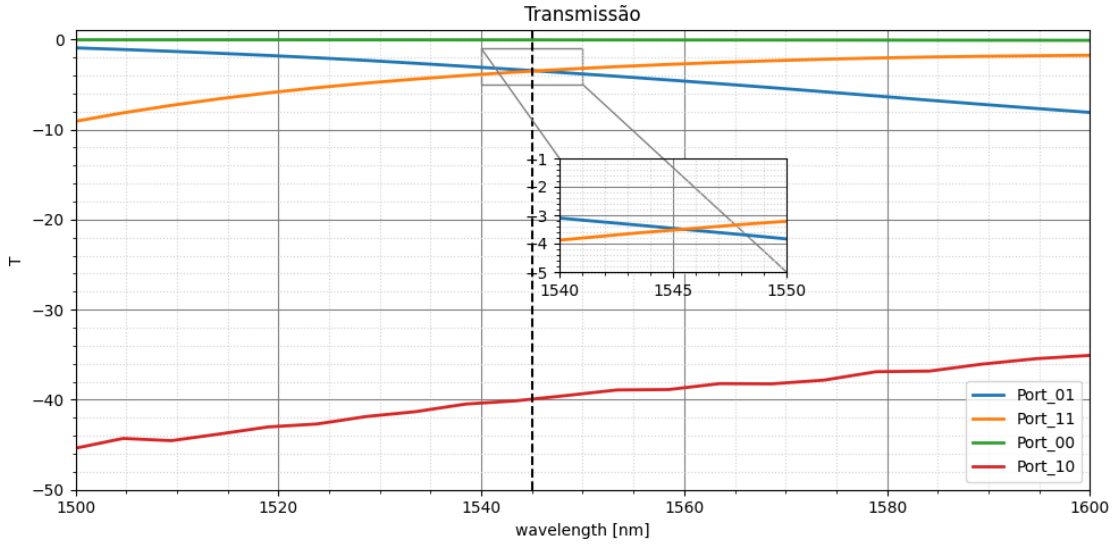



**Comprimento de Acoplamento para cada percentual de potencia tranferida**

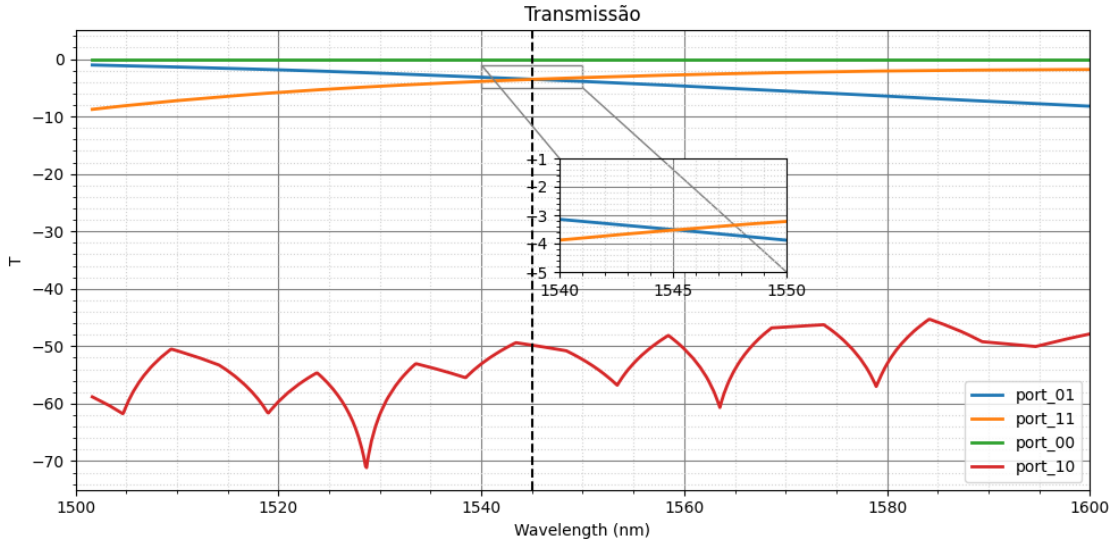
Como pode ser visto no gráfico acima o comprimento para uma razão 50/50 não é de 10,53 microns como era no modo TE. Abaixo estão o comportamento do modo TM para o comprimento do modo TE




**Transmissão Otimizada – FDTD - Linear**




**Transmissão Otimizada – FDTD – Log**




**Transmissão Otimizada – INTERCONCT – Log**

# Referencias

[1] https://optics.ansys.com/hc/en-us/categories/360001998954-Scripting-Language

[2] https://developer.ansys.com/docs/lumerical/python-lumapi

[3] https://optics.ansys.com/hc/en-us/articles/360042305334-Grating-coupler

[4] https://www.ansys.com/simulation-topics/what-are-s-parameters