Apodization of coupled cavity array for waveguide quantum electrodynamics: design and simulation

– Hybrid Quantum Circuit Laboratory (HQC) –

Travaux pratiques IV, Physics Master EPFL

submitted by

Daniele Cucurachi

January, 2022

Daniele Cucurachi
daniele.cucurachi@epfl.ch
Student ID: 328929

Supervisors
1st: Dr. Vincent Jouanny
2nd: Prof. Pasquale Scarlino

École polytechnique fédérale de Lausanne (EPFL)

Contents

1 Introduction and Motivations 1

2 Materials and Methods 1

2.1 Apodization in arrays of lumped-element resonators 2

2.2 Softwares and tools 4

2.3 Workflow 4

2.3.1 Optimal capacitance values determination 4

2.3.2 LTspice/Cadence simulations and design code 5

2.3.3 Design optimization with $Z_0 = 50\ \Omega$ 6

2.3.4 Design optimization with $Z_0 = 500\ \Omega$ 11

3 Results and Conclusions 12

3.1 Results 12

3.2 Conclusions 13

A Appendix 16

1 Introduction and Motivations

In the present work, we aim to optimize the design of a coupled resonator waveguide QED to obtain flat transmission on a defined frequency range through apodization of the transmission peaks. In order to reach apodization we slightly modify the physical parameters of the resonators at both ends of the waveguide device so that the reflections cancel out the Fabry–Perot resonances at the resonant frequencies. This leads to a broadening of the characteristic peaks usually observed in transmission for coupled cavity arrays. Under optimal conditions, investigated by Chak et al. [1], we can exploit these broadenings to obtain flat transmission on a frequency band whose width depends on the number of resonators in the waveguide and on their coupling strength.

Apodized coupled resonator waveguides have been used for filtering applications as they operate as band-pass filters [2]. However, in recent years these devices have captivated more interest because of their possible use in slow light applications [3]. Slow light devices are based on the idea that the dispersion relation in periodic structures flattens at the transmission band edges. Since the group velocity is defined by the derivative of the dispersion relation, it is therefore small in regions where the dispersion relation goes flat, meaning that around the transmission band edges we can achieve both large transmission and low group velocity. These devices can be used to study non-Markovian dynamics i.e. dynamics of physical systems with memory of the past interactions [4].

Superconducting coplanar waveguide resonators have been also used to achieve strong coupling regime, paving the way for a multitude of new investigations [5, 6]. In addition, they represent an ideal platform to study the quantum dynamics of systems interacting with a continuum of EM modes (multimode coupling).

2 Materials and Methods

In the following section, I am going to provide a detailed account of the procedure that was followed in completing the project and the tools that were adopted. After a concise review of the theoretical background (subsection 2.1) I will present the software tools used for simulations (subsection 2.2) and the workflow (subsection 2.3) that guided the entire project, reporting all the relevant remarks and observations.

Note that in Appendix A is reported the Python code used for the resonator waveguide design.

2.1 Apodization in arrays of lumped-element resonators

We consider an array of $n = 8$ resonators, all featuring the same parameters, except for the first two and the last two.

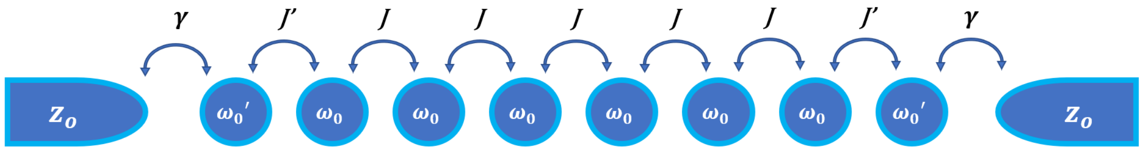


Figure 1: *Coupled resonators waveguide schematic. ω_0 and ω_0' are the resonators resonant frequencies while J and J' are the coupling strengths. γ is the energy decay rate of the edge sites*

We describe the resonator array using the tight-binding formalism, assuming periodic boundary conditions and considering only nearest-neighbor coupling. The corresponding n-site Hamiltonian is the following:

$$H = \begin{pmatrix} \omega_0' + i\gamma & \text{J'} & 0 & & & & \\ \text{J'} & \omega_0 & \text{J} & & & & \\ 0 & \text{J} & \omega_0 & & & & \\ & & & \cdots & & & \\ & & & & \omega_0 & \text{J} & 0 \\ & & & & \text{J} & \omega_0 & \text{J'} \\ & & & & 0 & \text{J'} & \omega_0' + i\gamma \end{pmatrix}$$

Where ω_0' is the first (and last) resonator resonant frequency, J is the inter-site coupling, J' is the inter-site coupling between the first two (and the last two) resonators, γ is the energy decay rate of the edge sites due to the coupling to the measurement setup.

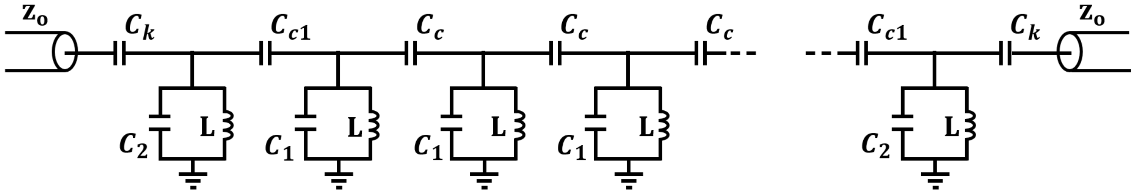


Figure 2: *Equivalent circuit representing the resonators waveguide on lumped element description. Note that we consider only nearest-neighbor coupling, secondary coupling has been neglected.*

We realize this Hamiltonian with a lumped-element resonator array coupled to transmission lines (Z_0 feedline input impedance).

The lumped-element resonator array parameters are related to the Hamiltonian parameters by:

$J = \frac{C_c \omega_0}{2C_\Sigma}$: Inter-site coupling with total capacitance $C_\Sigma = C + 2C_c$ and resonant frequency $\omega_0 = \frac{1}{\sqrt{LC_\Sigma}}$

$J' = \frac{C_{c1}\omega_0'}{2C_{\Sigma'}}$: First two (and last two) resonators coupling with total capacitance $C_{\Sigma'} = C_k + 2C_2 + C_{c1}$ and resonant frequency $\omega_0' = \frac{1}{\sqrt{LC_{\Sigma'}}}$

$\gamma = \frac{C_k^2 Z_0 \omega_0}{C_\Sigma^2 Z_r}$: Energy decay rate of the edge sites, with $Z_r = \sqrt{L/C_\Sigma}$

In order to reach optimal apodization and obtain a flat passband in transmission, the conditions proposed by Chak and coworkers are the following [1]:

$$\omega_0' = \omega_0$$
$$J' = \sqrt{2}J$$
$$\gamma/2 = 2J sin(\psi)$$

where $\psi = \pi/2$ (best matching at the middle of the band) or $\psi = \pi/4$ (best matching at the band edges). Considering the limit $C > C_k >> C_{c1}, Cc$, we can meet these conditions with the following capacitances values:

$$C_{c1} = \sqrt{2}C_c$$
$$C_k = \sqrt{\frac{2C_c sin(\psi)}{Z_0\omega_0}}$$
$$C_1 = C - (C_{c1} - C_c)$$
$$C_2 = C - (C_{c1} + C_k - 2C_c)$$

Where the first two conditions optimize the capacitance values to have the best result, while the last two ensure that ω_0'=ω_0.

In section 3 LTspice simulations have been used to prove that the values derived through these equations give indeed the best result: even with a small number of resonators, flat transmission is observed on a well-defined frequency range proportional to the number of resonators.

2.2 Softwares and tools

In the current subsection, I am going to briefly present the software used for simulations and the Python library used to design the resonators waveguide.

LTspice and Cadence Microwave Office: circuit design softwares. High-performance SPICE simulation software, schematic capture and waveform viewer with enhancements and models for easing the simulation of analog circuits.

Sonnet Software: EDA software solutions. Note that, despite Sonnet and LT-spice have been used to calculate the transmission spectrum, LTspice simulations are based on a circuit diagram that models the device while Sonnet simulates by numerically calculating Maxwell equations point by point on a 2-D representation of the real design. for high-frequency RF/MW electromagnetic analysis.

ANSYS: 3D electromagnetic (EM) simulation software for designing and simulating high-frequency electronic devices.

gdspy library: Python module that allows the creation of GDSII stream files for micro and nanofabrication.

2.3 Workflow

The following subsection is dedicated to the presentation of the project workflow: I will provide an overview of the resonator waveguide design process, starting from the equations and finishing with the final waveguide design that will be fabricated.

2.3.1 Optimal capacitance values determination

The first step consisted in exploiting the equations, derived in subsection 2.1, in order to find the best capacitance values for the resonator waveguide to obtain a flat band in transmission. Since we have more parameters than equations: 6 parameters to determine

$$C, C_c, C_{c1}, C_k, C_1, C_2$$

and only 4 equations,

$$C_{c1} = \sqrt{2}C_c$$

$$C_k = \sqrt{\frac{2C_c sin(\psi)}{Z_0 w_0}}$$

$$C_1 = C - (C_{c1} - C_c)$$

$$C_2 = C - (C_{c1} + C_k - 2C_c)$$

we are free to assign arbitrary values to two of these parameters and successively derive the others. This comes really in handy when considering that, in circuit quantum electrodynamics, we have a narrow range of possible values for these capacitances due to the physical limits of the devices. We can therefore exploit these degrees of freedom to choose some convenient values that are easy to realize.
Note that, in the present analysis, we are not considering L_0 and Z_0 since these parameters are fixed. Z_0 is the input feedline impedance for which we have only two possible values: 50Ω, and 500Ω which can be reached by tapering. On the other hand, we take L_0 as a fixed parameter ($L_0 = 38.8nH$) because we want a large inductance since it is needed for metamaterial waveguides [7, 8, 9], paving the way for future applications of the devices designed in this work. When increasing L_0 we induce a redshift in the resonators' resonant frequency (defined as $w_0 = \frac{1}{\sqrt{LC_\Sigma}}$), therefore we cannot use inductance values larger than $\approx 40\ nH$, otherwise the resonant peaks in transmission would appear in a range of the frequency spectrum we are not able to probe with the available lab instruments.

I started with $Z_0 = 50\Omega$ and chose as starting point the following values:

$$C_c = 0.43\ fF \quad , \quad C = 25\ fF$$

where C_c represent the coupling capacitance between two neighboring resonators in the waveguide and C is the coupling capacitance between a single resonator and ground since these values have been already obtained for other devices in the past, therefore we knew they were achievable in practice.

Using Chak's equations we successively derived the values for the other parameters:

C_c	C	C_k	C_{c1}	C_1	C_2
0.43 fF	25 fF	19.63 fF	0.61 fF	24.82 fF	5.62 fF

Table 1: *Values derived through Chak's equations starting from* $C_c = 0.43\ fF$, $C = 25\ fF$

2.3.2 LTspice/Cadence simulations and design code

Once a proper choice of the starting parameters has been made, I simulated the transmission spectrum of the resonator waveguide using LTspice and Cadence Microwave Office. Simulations and parametric analysis have been used to confirm that the derived parameter values were indeed the ones giving the best results. As it is shown in Figure 4, the closer the parameters are to the derived ones, the flatter is the band we observe in transmission.

After the simulations confirmed this starting set of values, I designed the first prototype of the device. In order to optimize and speed up the designed process I coded a script in Python, using the *gdspy* library, that prints out a fully customizable gds file of the resonators waveguide (the Python code is reported in Appendix A)

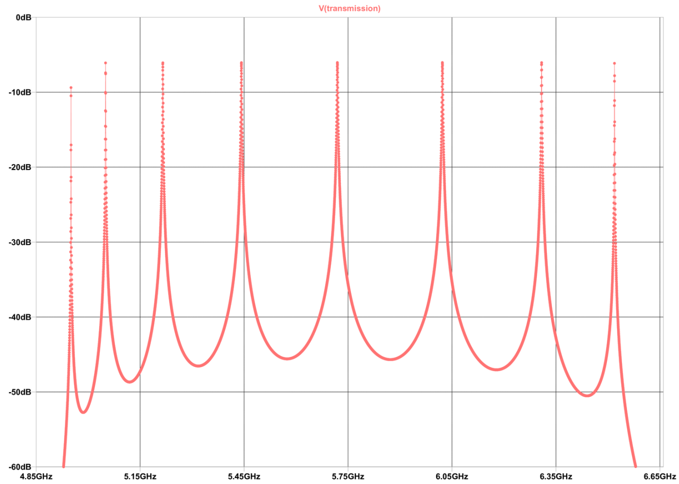


Figure 3: *Transmission vs frequency for a standard coupled resonators waveguide not apodized (the parameters are set as: $C_c = C_{c1} = C_k$ and $C_1 = C_2 = C$).*

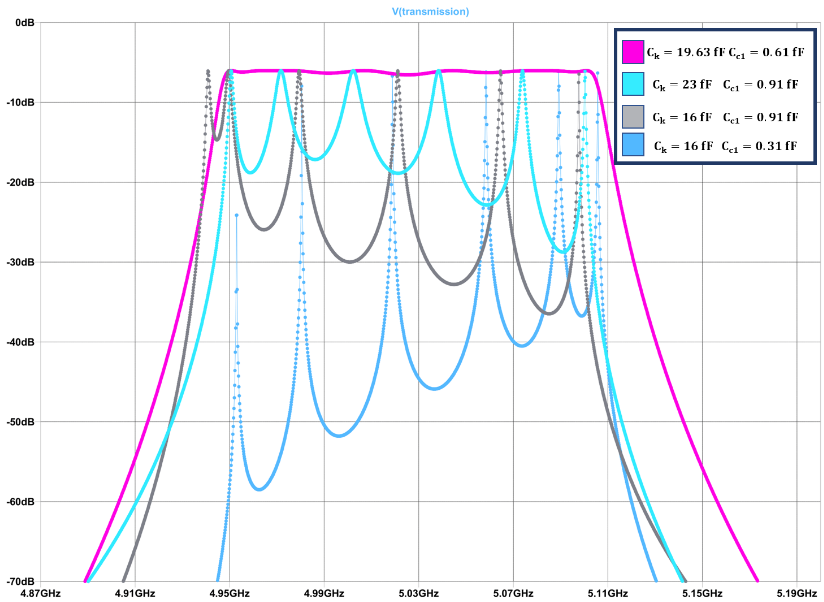


Figure 4: *Transmission spectrum for the apodized coupled resonators waveguide. The 4 traces in different colors represent 4 different sets of values for the parameters C_k and C_{c1}. It is clear that the best result is achieved when using the values derived from Chak's equations: for $C_k = 19.63$ fF and $C_{c1} = 0.61$ fF a flat band is observed in transmission.*

2.3.3 Design optimization with $Z_0 = 50\ \Omega$

The gds file has been then used to run a simulation on ANSYS to numerically calculate the capacitance matrix of this first design. The capacitance matrix provides, for each element of the waveguide (single resonators, ground, feedline capacitors), the coupling capacitances with respect to the other elements. The capacitance matrix values have been then compared to the desired capacitance values derived from the equations. This first comparison allowed me to understand how far the starting design was from the optimal one I was looking for.

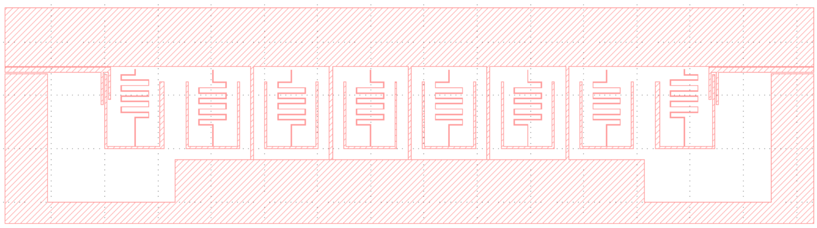


Figure 5: *First waveguide design: the ground piece between the first (last) and second (last second) resonators has been removed to reduce the capacitance to ground for the first resonator (C_2). Note that, for ANSYS simulations, the resonators must be separated from the ground (as you can see in this picture) so that they can be simulated as individual elements.*

	C_c	C	C_k	C_{c1}	C_1	C_2
Desired cap values	0.43 fF	25 fF	19.63 fF	0.61 fF	24.82 fF	5.62 fF
First design cap values	≈1.5 fF	≈17 fF	≈5.5 fF	≈2.5 fF	≈15 fF	≈14.5 fF

Table 2: *Comparison between ANSYS simulation results and optimal capacitance values derived from Chak's equation*

The next phase has been characterized by a trial and error approach aimed at understanding how different modifications in the physical structure of the device affected the capacitance values. To this purpose, I have run several simulations on ANSYS changing each time a different parameter in the waveguide: for example capacitors thickness, capacitors length, distances between resonators and ground, and so on.

This analysis showed that, as expected, we can modify the coupling capacitance between two elements in the waveguide by:

- Increasing or decreasing the distance between the two elements. However, we cannot have a too large separation since we would have problems with the lithography process (used to fabricate the waveguide): first of all, it would imply longer time and higher costs for the process, and it would also increase the probability of errors in the making.
- Modifying the thickness of the two elements. However, we cannot decrease it too much or an element that is supposed to be a capacitor will start behaving as an inductor in practice.
- Modifying the exposed surface between the two elements (see example in Figure 6).

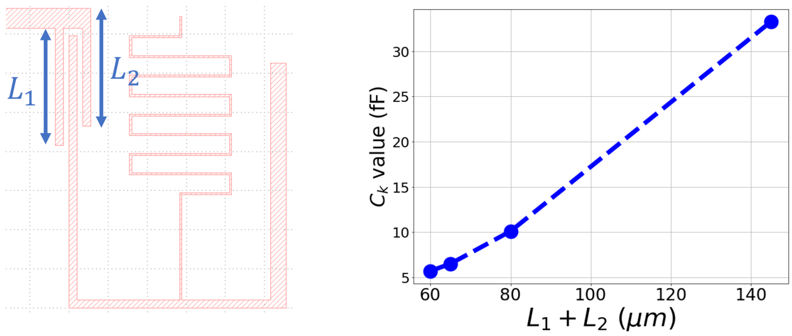


Figure 6: *Impact of the feedline capacitor length on C_k value. C_k represents the coupling capacitance between the first resonator and the feedline capacitor. Note that there are other waveguide features that strongly affect the value of C_k: the distance between the feedline capacitor and the resonator capacitor, and the thickness of the capacitors.*

After this initial study, I started modifying the waveguide design in order to achieve the optimal capacitance values. This step by step process gradually converged towards the optimal waveguide design: at each step, the design was modified and simulated with ANSYS, then the simulations' results, compared with the desired capacitance values, provided feedback that suggested how to further modify the design.

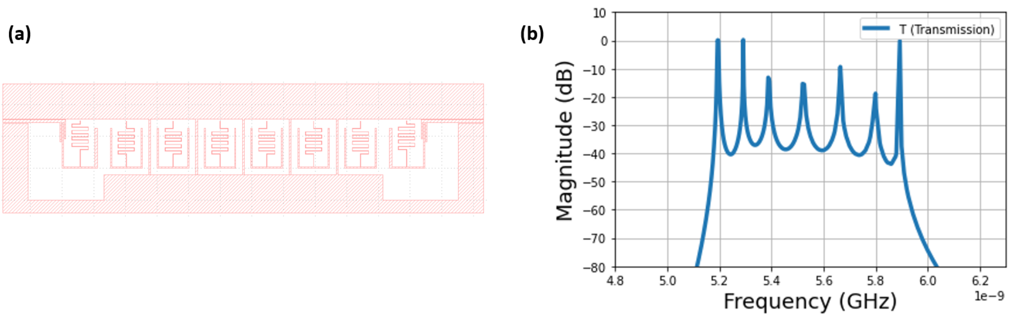


Figure 7: *Sonnet simulation of the first waveguide design. The transmission spectrum shows several well defined peaks meaning that the waveguide is not apodized.*

As shown Figure 5, the ground between the first and second resonators has been removed in order to decrease the capacitance to ground for the first resonator with respect to the others (indeed for the optimal capacitance values we have $C_2 << C$) and increase the capacitive coupling between the first two resonators (since $C_{c1} > C_c$).

During this design optimization process, together with ANSYS simulations, Sonnet simulations (Figure 7) were used to monitor the evolution of the transmission spectrum.

Thanks to this process I managed to design a waveguide that met the optimal parameters for C, C_c, C_{c1}, C_k, C_1. However, the simulations showed that, for this kind of device, obtaining C_2 =5.62 fF was not feasible (where C_2 is the coupling capacitance between the first/last resonators and ground). Even pushing the design to the limits the C_2 value calculated by ANSYS did not get smaller than $\approx 10-11$ fF (Figure 8). I could not even lower the values of C_c and C_{c1} (where $C_{c1} = \sqrt{2}C_c$) to increase the optimal value of C_2 (where $C_2 = C - (C_{c1} + C_k - 2C_c)$) because they both have a very narrow range of achievable values ($\approx 0.1 - 3$ fF) and they were already quite small. If C_c and C_{c1} are too small, the waveguide does not work properly since these two values are proportional to the capacitive coupling between the resonators $J \propto C_c$ and $J' \propto C_{c1}$.

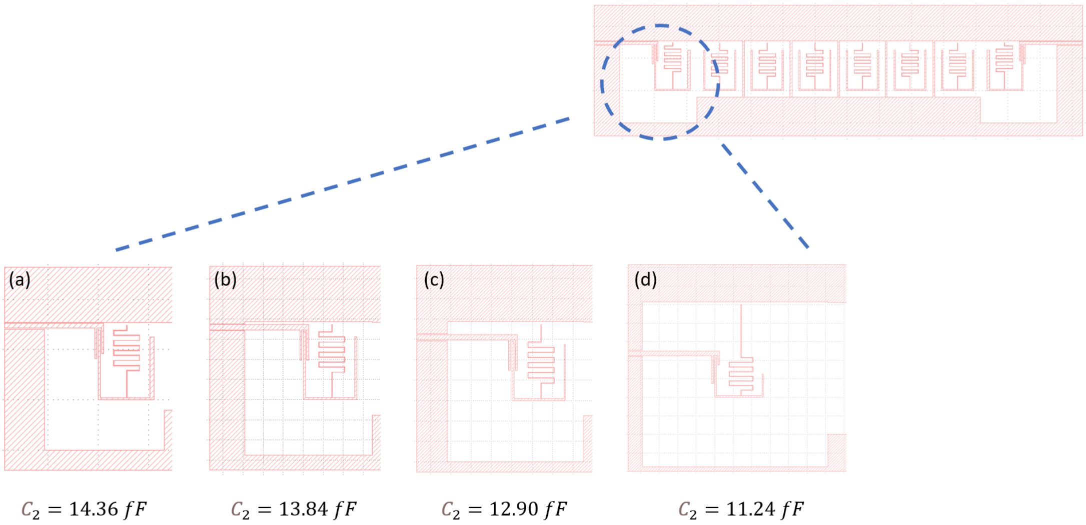


Figure 8: *Attempts to decrease C_2 (first and last resonators' capacitance to ground) in order to achieve the optimal value. In designs (a) and (b) the separation between the first resonator and the ground is increased, in (c) and (d) the separation is increased and the first resonator capacitor thickness is reduced. None of the proposed designs reach the optimal value for C_2 and it is not possible to increase further the distance from the ground since this would give problems during fabrication with the lithography process.*

In order to overcome this problem, the solution was finding a new optimal capacitance values set. Therefore I exploited again the equations presented in subsection 2.1, imposing $C_2 = 13$ fF this time. Note that C_2 has to be kept as small as possible because it is directly related to C value: following Chak's equations, the larger C_2 the larger C is gonna be. Already for $C_2 \approx$15-20 fF, C reaches values that are not achievable in practice for this kind of devices. Imposing $C_2 = 13$ fF allows to have achievable values for both C_2 and C.

C_c	C	C_k	C_{c1}	C_1	C_2
1.00 fF	48.00 fF	35.29 fF	1.41 fF	47.58 fF	13.30 fF

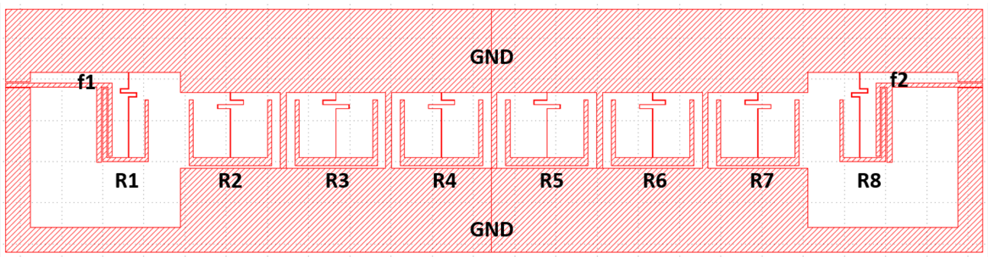
Table 3: *New set of optimal values derived starting from $C_2 = 13$ fF and $C_c = 1$ fF.*

Starting from $C_2 = 13$ fF and $C_c = 1$ fF, which ensures a good coupling between the resonators, I derived the new set of optimal capacitance values reported in Table 3.

This new set contains all achievable capacitance values. However, in this case we have the resonators' resonant frequencies approaching $4GHz$:

$$f_0 = \frac{1}{\sqrt{LC_\Sigma}2\pi} \approx 3.6\ GHz$$

This is a problem for us because we cannot probe this range of frequencies with the instruments available in our lab.



	R1	R2	R3	R4	R5	R6	R7	R8	f1	f2	GND
R1	35.90 fF	1.52 fF	0.11 fF	0.054 fF	0.036 fF	0.027 fF	0.026 fF	0.022 fF	21.83 fF	0.021 fF	12.24 fF
R2	1.52 fF	35.44 fF	1.27 fF	0.11 fF	0.05 fF	0.03 fF	0.03 fF	0.026 fF	0.42 fF	0.026 fF	31.93 fF
R3	0.11 fF	1.27 fF	36.91 fF	1.24 fF	0.09 fF	0.043 fF	0.035 fF	0.027 fF	0.08 fF	0.03 fF	33.96 fF
R4	0.05 fF	0.11 fF	1.24 fF	36.87 fF	1.24 fF	0.10 fF	0.05 fF	0.04 fF	0.05 fF	0.03 fF	33.95 fF
R5	0.04 fF	0.05 fF	0.09 fF	1.24 fF	36.86 fF	1.24 fF	0.11 fF	0.05 fF	0.03 fF	0.05 fF	33.95 fF
R6	0.03 fF	0.03 fF	0.04 fF	0.10 fF	1.24 fF	36.85 fF	1.27 fF	0.11 fF	0.03 fF	0.08 fF	33.90 fF
R7	0.03 fF	0.03 fF	0.035 fF	0.05 fF	0.11 fF	1.27 fF	35.32 fF	1.52 fF	0.03 fF	0.42 fF	31.83fF
R8	0.02 fF	0.03 fF	0.03 fF	0.04 fF	0.05 fF	0.11 fF	1.52 fF	36.13 fF	0.02 fF	22.06 fF	12.24 fF
f1	21.83 fF	0.42 fF	0.08 fF	0.05 fF	0.03 fF	0.03 fF	0.03 fF	0.02 fF	46.15 fF	0.02 fF	23.63 fF
f2	0.021 fF	0.03 fF	0.03 fF	0.03 fF	0.05 fF	0.08 fF	0.42 fF	22.06 fF	0.02 fF	46.50 fF	23.76 fF
GND	12.24 fF	31.93 fF	33.96 fF	33.95 fF	33.95 fF	33.90 fF	31.83fF	12.24 fF	23.63 fF	23.76 fF	271.4 fF

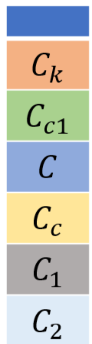


Figure 9: *ANSYS Capacitance matrix of the optimal design for the case* $Z_0 = 50\ \Omega$

The only remaining parameter that can be tuned is L_0, even though reducing L_0 means giving up on high inductance to ground for our device. We have that:

$$L = L_\square \frac{l}{w}$$

Where $L_\square$ is the inductance per square, l and w are the length and the width of the inductor wire. By shortening the inductor wire by almost half of its original length, I reduced L_0 from 38.8 nH to 20 nH, taking the resonator's resonant frequencies to ≈ 5 GHz.

With this new set of parameters, adopting the same optimization procedure described before, I obtained the design shown in Figure 9. Note that the thickness of the resonators' capacitors has been increased and the distance between the resonators and ground has been strongly decreased to achieve large C values. On the other hand, reaching a large C_k has been easy since the interdigitated feed line capacitor allows good control over this parameter.

Note also that, despite the capacitance values calculated through ANSYS simulation (shown in Figure 9) are not close to the optimal values (shown in Table 3), this design is the one giving the best results for the case $Z_0 = 50\ \Omega$.

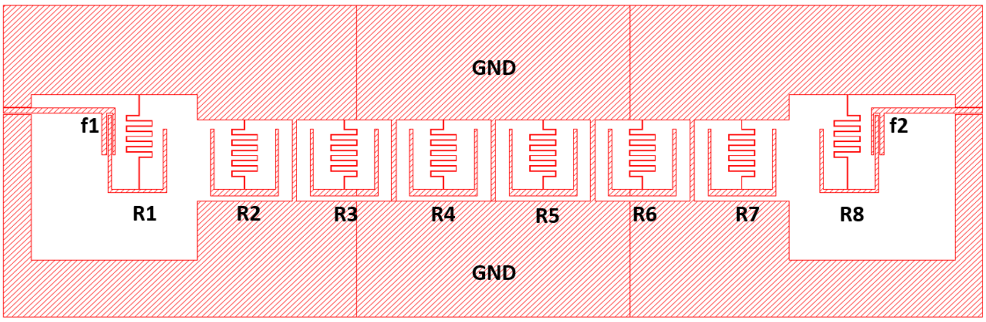
2.3.4 Design optimization with $Z_0 = 500\ \Omega$

It is possible to achieve an input impedance of $Z_0 = 500\ \Omega$ by tapering. This second case turned out to be much easier since a larger value of Z_0 allows to reach apodization with lower capacitance values with respect to the previous case, which is easier to achieve, while maintaining at the same time a large L_0 ($L_0 = 38.8$ nH).

Following the same process used for the $Z_0 = 50\ \Omega$ case, I derived the set of optimal values reported in Table 4.

C_c	C	C_k	C_{c1}	C_1	C_2
1.00 fF	22.70 fF	9.35 fF	1.41 fF	22.20 fF	13.98 fF

Table 4: *New set of optimal values for the case $Z_0 = 500\ \Omega$*



	R1	R2	R3	R4	R5	R6	R7	R8	f1	f2	GND
R1	25.18 fF	1.43 fF	0.10 fF	0.034 fF	0.017 fF	0.014 fF	0.01 fF	0.01 fF	9.97 fF	0.005 fF	13.58 fF
R2	1.43 fF	23.78 fF	1.01 fF	0.08 fF	0.03 fF	0.01 fF	0.01 fF	0.01 fF	0.13 fF	0.005 fF	21.04 fF
R3	0.10 fF	1.01 fF	24.50 fF	1.00 fF	0.08 fF	0.02 fF	0.01 fF	0.01 fF	0.03 fF	0.006 fF	22.21 fF
R4	0.034 fF	0.08 fF	1.00 fF	24.19 fF	0.97 fF	0.07 fF	0.03 fF	0.02 fF	0.01 fF	0.01fF	21.95 fF
R5	0.02 fF	0.03 fF	0.08 fF	0.97 fF	24.08 fF	0.99 fF	0.08 fF	0.03 fF	0.01 fF	0.01 fF	21.85 fF
R6	0.01 fF	0.01 fF	0.02 fF	0.07 fF	0.99 fF	24.43 fF	1.00 fF	0.1 fF	0.005 fF	0.03 fF	22.17 fF
R7	0.01 fF	0.01 fF	0.01 fF	0.03 fF	0.08 fF	1.00 fF	23.47 fF	1.41 fF	0.005 fF	0.13 fF	20.77 fF
R8	0.01 fF	0.01 fF	0.01 fF	0.02 fF	0.03 fF	0.1 fF	1.41 fF	25.24 fF	0.005 fF	9.91 fF	13.71 fF
f1	9.97 fF	0.13 fF	0.03 fF	0.01 fF	0.01 fF	0.005 fF	0.005 fF	0.005 fF	55.63 fF	0.003 fF	45.45 fF
f2	0.005 fF	0.005 fF	0.006 fF	0.01fF	0.01 fF	0.03 fF	0.13 fF	9.91 fF	0.003 fF	55.42 fF	45.30 fF
GND	13.58 fF	21.04 fF	22.21 fF	21.95 fF	21.85 fF	22.17 fF	20.77 fF	13.71 fF	45.45 fF	45.30 fF	248.07 fF

C_k

C_{c1}

C

C_c

C_1

C_2

Figure 10: *ANSYS Capacitance matrix of the optimal design for the case $Z_0 = 500\ \Omega$*

Again, we obtained the design reported in Figure 10 for this new case by gradual design optimization following the steps described previously.

3 Results and Conclusions

In the following section, the final results and the conclusions are presented: I will show the two final designs, products of the optimization process presented in subsection 2.3, and the relative transmission spectra.

3.1 Results

The optimal design for the case $Z_0 = 50\ \Omega$ is reported in Figure 11:

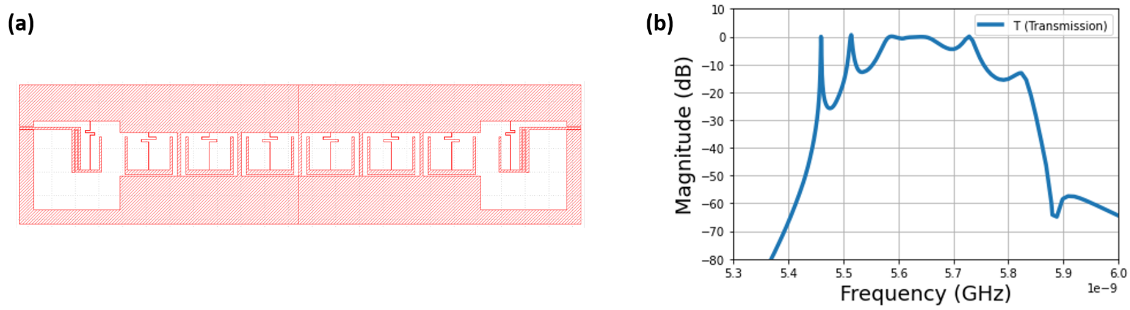


Figure 11: *Optimal design* ***(a)*** *and transmission spectrum* ***(b)*** *for the case* $Z_0 = 50\ \Omega$

For the second case, $Z_0 = 500\ \Omega$, the results are shown in Figure 12:

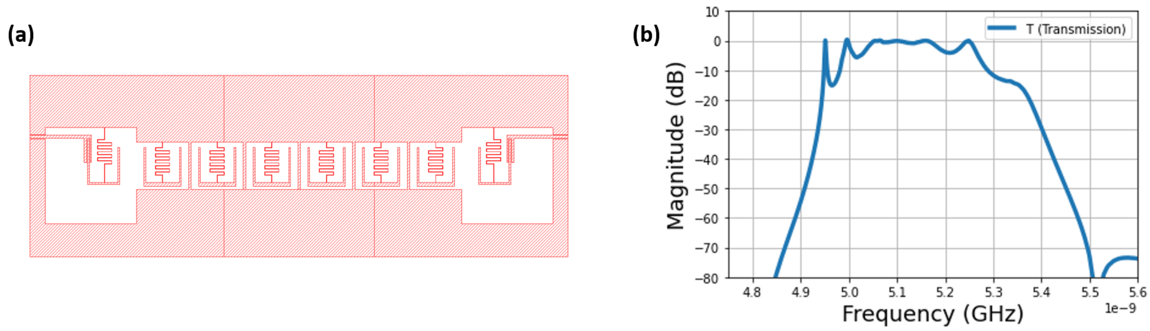


Figure 12: *Optimal design* ***(a)*** *and transmission spectrum* ***(b)*** *for the case* $Z_0 = 500\ \Omega$

In Figure 11 and Figure 12 we can observe that both the transmission spectra feature an asymmetric shape, a shoulder is visible on the right side of the bands. This asymmetry is inherited from the resonator waveguide transmission spectrum, which already shows this characteristic when not apodized (see Figure 3).
Another common characteristic is the presence of two residual peaks on the left side of the bands. We observe that these peaks are less pronounced in the second case ($Z_0 = 500\ \Omega$), where we also have a larger frequency range with flat transmission: ≈ 0.3 GHz, compared to ≈ 0.15 GHz for the $Z_0 = 50\ \Omega$ case. In order to mitigate these oscillations at the band edges we could use a larger number of resonators: with

more resonators we can increase the extent of the band and obtain a flat transmission on a larger frequency range.

In Figure 13 are pictured the two different resonator designs for the two waveguides. In the first case ($Z_0 = 50\ \Omega$) the capacitor has a wider and thicker bottom part in order to reach a larger value for the capacitance to ground $C = 48$ fF, while $C = 22.7$ fF in the case $Z_0 = 500\ \Omega$. On the other hand, in order to compensate this larger capacitance value, the inductor wire is reduced by half in this case so that the resonators resonant frequencies (defined as $f_0 = \frac{1}{2*\pi\sqrt{LC_\Sigma}}$, where C is the dominant factor in $C_\Sigma = C + 2C_c$) falls into the range 4.5 GHz - 6 GHz.

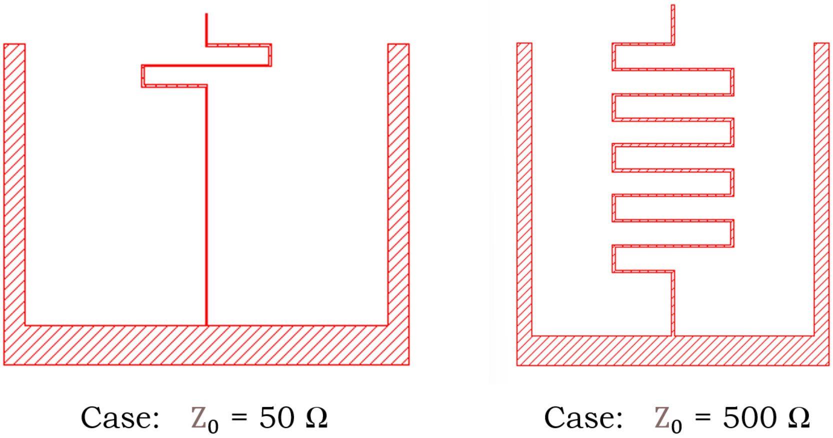


Figure 13: *Comparison between the two different resonators designs for the two waveguides*

3.2 Conclusions

Starting from the equations derived in subsection 2.1, we successfully designed two coupled resonators waveguides, with different input impedances $Z_0 = 50\ \Omega$ $Z_0 = 500\ \Omega$, optimized to achieve apodization. Both devices clearly show flat transmission respectively in the frequency ranges $5.6 - 5.8$ GHz ($Z_0 = 50\ \Omega$ case) and $5 - 5.3$ GHz ($Z_0 = 500\ \Omega$ case), according to Sonnet simulations. These frequency ranges are optimal to study waveguide-SC qubits multimode coupling since superconducting qubits usually have frequencies in the range $4 - 8$ GHz [10, 11, 12]. The waveguide designs have been derived through the process explained in subsection 2.3. In order to optimize and speed up the design process we developed a Python program that prints out a fully customizable gds file of the resonators waveguide (see Appendix A).

References

[1] Philip Chak and J. E. Sipe. "Minimizing finite-size effects in artificial resonance tunneling structures". In: *Opt. Lett.* 31.17 (Sept. 2006), pp. 2568–2570. DOI: 10.1364/OL.31.002568. URL: http://www.osapublishing.org/ol/abstract.cfm?URI=ol-31-17-2568.

[2] J. Capmany et al. "Apodized coupled resonator waveguides". In: *Optics express* 15 (Sept. 2007), pp. 10196–206. DOI: 10.1364/OE.15.010196.

[3] Simone Gasparinetti Simon Mathis Jean-Claude Besse. "Microwave Waveguides with Engineered Dispersion based on Arrays of Lumped-Element Resonators". In: *ETH Zurich, semester project* (Oct. 2017).

[4] Vinicius S. Ferreira et al. "Collapse and Revival of an Artificial Atom Coupled to a Structured Photonic Reservoir". In: *Phys. Rev. X* 11 (4 Dec. 2021), p. 041043. DOI: 10.1103/PhysRevX.11.041043. URL: https://link.aps.org/doi/10.1103/PhysRevX.11.041043.

[5] Abdufarrukh A. Abdumalikov et al. "Vacuum Rabi splitting due to strong coupling of a flux qubit and a coplanar-waveguide resonator". In: *Phys. Rev. B* 78 (18 Nov. 2008), p. 180502. DOI: 10.1103/PhysRevB.78.180502. URL: https://link.aps.org/doi/10.1103/PhysRevB.78.180502.

[6] Andreas Wallraff et al. "Strong coupling of a single photon to a superconducting qubit using circuit quantum electrodynamics". In: *Nature* 431 (Oct. 2004), pp. 162–7. DOI: 10.1038/nature02851.

[7] D. J. Egger and F. K. Wilhelm. "Multimode Circuit Quantum Electrodynamics with Hybrid Metamaterial Transmission Lines". In: *Phys. Rev. Lett.* 111 (16 Oct. 2013), p. 163601. DOI: 10.1103/PhysRevLett.111.163601. URL: https://link.aps.org/doi/10.1103/PhysRevLett.111.163601.

[8] C. Caloz and T. Itoh. "Transmission line approach of left-handed (LH) materials and microstrip implementation of an artificial LH transmission line". In: *IEEE Transactions on Antennas and Propagation* 52.5 (2004), pp. 1159–1166. DOI: 10.1109/TAP.2004.827249.

[9] H. Wang et al. "Mode Structure in Superconducting Metamaterial Transmission-Line Resonators". In: *Phys. Rev. Applied* 11 (5 May 2019), p. 054062. DOI: 10.1103/PhysRevApplied.11.054062. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.11.054062.

[10] Xiu Gu et al. "Microwave photonics with superconducting quantum circuits". In: *Physics Reports* 718-719 (July 2017), pp. 1–102. DOI: 10.1016/j.physrep.2017.10.002.

[11] Morten Kjaergaard et al. "Superconducting Qubits: Current State of Play". In: *Annual Review of Condensed Matter Physics* 11.1 (Mar. 2020), pp. 369–395. ISSN: 1947-5462. DOI: 10.1146/annurev-conmatphys-031119-050605. URL: http://dx.doi.org/10.1146/annurev-conmatphys-031119-050605.

[12] P. Krantz et al. “A quantum engineer’s guide to superconducting qubits”. In: *Applied Physics Reviews* 6.2 (June 2019), p. 021318. ISSN: 1931-9401. DOI: 10.1063/1.5089550. URL: http://dx.doi.org/10.1063/1.5089550.

A Appendix

Python design code

```
# -*- coding: utf-8 -*-
"""
@author: Daniele Cucurachi

FULLY CUSTOMIZABLE COUPLED RESONATORS WAVEGUIDE GDS DESIGN

    the function is divided into 4 parts, each part is related to the design of a specific component. In the end, all the
    components are put together to crate the full WG. We have in order:

        1) standard resonators pairs: the resonators pairs that form the waveguide

        2) first resonator (+ feedline capacitor): first resonator in the waveguide and input feedline capacitor

        3) last resonator: mirrored copy of the first resonator

        4) ground design


     NOTE: the various parts of the function create the negatives of the components designs we need. At the end, the boolean "not" function is used to subtract these negatives from a rectangle (which is the ground) and we obtain the full WG

"""


#%% # -------------------------------------------------

# IMPORT LIBRARIES

import  numpy as np
import  gdspy


#%% # -------------------------------------------------



def RES_WG_APO():

```

```python

    # define all the parameters

    # NOTE: the default dimensions are micrometers (um)


    """1) STANDARD RESONATORS PAIRS"""

    # STD INDUCTOR

    L = 279 # total length of the inductor wire
    s = 4.5 # interspacing between the inductor windings
    wid = 0.5 # width of inductor wire

    compact = False # chose if compact inductor design or not

    # STD CAPACITOR

    A = 62 # horizontal dimension of U-capacitor
    t = 3 # thickness of U-capacitor
    add_t = 3 # added thickness for the bottom capacitor segment

    # RESONATOR GROUND DISTANCES

    dist_ground = 5 # distance between the capacitor and the ground
    res_dist_l = 29 # left lateral distance between two resonators in the resonators chain
    res_dist_r = 29 # right lateral distance between two resonators in the resonators chain
    inter_cell_dist = 4 # space between the two cells containing the resonators
    ind_to_ground = 10 # last inductor segment that connects the inductor to ground (inter_res_dist)
    centre=(0,0) # centre where I start to draw the design


    """2) FIRST RESONATOR + FEEDLINE CAPACITOR"""

    # F INDUCTOR

    Li = 279 # total length of the inductor wire
    si = 4.5 # interspacing of turns of the inductor wire
    wi = 0.5 # width of inductor wire
    d_second =20 # last segment of the inductor (connection to ground. It defines the distance from ground at the top of the resonator cell)

```

```
    compact_ind = False # chose if compact inductor design or not
    separated_elements = False # chose if you want to separate the inductor from the GND (for ANSYS simulations)
    separation_length = 2 # separation length for separated elements option

    # FEEDLINE INTERDIGITATED CAPACITOR

    w2 = 2 # external feedline capacitor width
    sc = 1.5 # lateral separation between the first resonator capacitor and the feedline capacitor
    y = 2 # vertical separation between the first resonator capacitor and the feedline capacitor
    w = 5 # feedline width
    w1 = 4 # width of the internal capacitor arm
    L2 = 45 # length of the internal capacitor arm
    L1 = 40 # length of the external capacitor arm
    Bwg = 200 # feedline length

    # F CAPACITOR

    Ac = 50 # horizontal dimension of U-capacitor (without additions, the final effective width is = A+Ws-t+Wl)
    tc = 3 # thickness of U-capacitor (note that the thickness of the capacitor wide arm can be tuned individually)
    Wl = t # wide arm thickness
    Lw = 60 # wide arm length
    Ws = (sc + w2) # thin arm shift

    # GROUND DISTANCES

    dist_ground_first = 65 # distance between capacitor and ground
    first_dist_lateral_l = 65 # left lateral distance between the capacitor and ground
    first_dist_lateral_r = 28 # right lateral distance between the capacitor and ground
    WG_width = 7 # feedline capacitor width

    # define two parameters that will be used when creating the ground geometry

    Lateral_l = Bwg + w1 + sc
    Lateral_r = Ws + Ac + Wl - tc + first_dist_lateral_r



    """3) LAST RESONATOR"""
```

```
    """4) GROUND DESIGN"""

    Height = 250 # distance between the first cell and the edge of the
    ↪  ground piece
    N_pairs = 3 # number of cells pairs in the waveguide (NOT including
    ↪  the first and last cells)



    """ANSYS/SONNET SIMULATION ADDED CAPACITOR THICKNESS """

    # the parameter T1 offer the possibility to increase the thickness
    ↪  of the second and last second capacitors
    # the parameter T2 offer the possibility to increase the thickness
    ↪  of the ground piece below  the second and last second capacitors

    # T1 and T2 are used to tune the capacitance to ground of the second
    ↪  and last second capacitors indipendently from the resonators in
    ↪  the waveguide

    T1 = 0# thickness of the added capacitor piece

    T2 = 0# thicnkess of the added ground piece




# -------------------------------------------------

    # RESONATORS WAVEGUIDE DESIGN: PART 1) 2) 3) 4)




    """1) STANDARD RESONATORS PAIRS"""


    # CHECK FOR DIMENSIONS MISMATCH IN THE DISTANCES BETWEEN THE
    ↪  RESONATORS (STD RESONATORS CHAIN)

    while (res_dist_l <= inter_cell_dist):
```

```
        print("MISMATCH: res_dist_l (lateral distance between two resonators in the chain) is equal or smaller than inter_cell_dist (width of the ground in between two resonators cells)")
        new = input("Enter new value for res_dist_l:")
        res_dist_l = float(new)


    while (res_dist_r <= inter_cell_dist):
        print("MISMATCH: res_dist_r (lateral distance between two resonators in the chain) is equal or smaller than inter_cell_dist (width of the ground in between two resonators)")
        new = input("Enter new value for res_dist_r:")
        res_dist_r = float(new)

    dist_lateral_l = (res_dist_l - inter_cell_dist)/2
    dist_lateral_r = (res_dist_r - inter_cell_dist)/2



    # PARAMETERS CALCULATIONS

    # horizontal dimension of the resonators pairs (these parameters will be used later to draw the waveguide geometry)

    Horizonta = 2*A + 2*dist_lateral_l + 2*dist_lateral_r + inter_cell_dist

    half_res = A/2 + dist_lateral_l

    #calculate horizontal dimension of inductor

    b = 3/5*A - 2*t

    #calculate number of windings in the inductor

    if compact == False:
        N = int((L-(ind_to_ground + (s+wid)))/(b+s))
    else:
        N = round((L-(ind_to_ground + (s+wid)))/(b+s))
    # print('N: ',N)

    #adapt first segment of the inductor

    d_prime = (L - (N*(s+b)+ind_to_ground)) + wid

```

```
    #calculate vertical dimension of capacitor

    B = N*(s+wid) + d_prime + t

    # define Ushape, set of points that will define the capacitor
    ↪  geometry

    U=[]
    x0, y0 = centre[0],centre[1]
    U.append([x0, y0])

    x1, y1 = x0, y0 - B + t/2
    U.append([x1, y1])

    x2, y2 = x1 + A - t, y1
    U.append([x2, y2])

    x3, y3 = x2, y2 + B - t/2
    U.append([x3, y3])

    U = np.asarray(U)
    centring = (A-t)/2, -B
    U = U - centring

    # additional thickness for the bottom segment of the capacitor

    bottom_cap = gdspy.Rectangle((-A/2,-add_t),(+A/2,0))

    #set of points that will define the meander (inductor) starting from
    ↪  centre of U

    M = []
    v0,w0 = centre[0], centre[1]+t
    M.append([v0,w0])
    v1,ww1 = v0, w0 + d_prime - wid/2
    M.append([v1,ww1])

    for i in range(N):
        if i%2 == 0:
            v2, ww2 = M[-1][0] - b/2 + wid/2, M[-1][1]
            M.append([v2,ww2])
            v3, w3 = v2, ww2 + (s+wid)
            M.append([v3,w3])
            v4, w4 = v3 + (b/2 - wid/2), w3
            M.append([v4,w4])
            # print(i)
        if (i+1)%2==0:
```

```python
            v5,w5 = M[-1][0] + b/2 - wid/2, M[-1][1]
            M.append([v5,w5])
            v6, w6 = v5, w5 + (s+wid)
            M.append([v6,w6])
            v7,w7 = v6 - b/2 + wid/2, w6
            M.append([v7,w7])
            # print(i)
        if (i+1)==N:
            # print('end turns:', M[-1])
            v8,w8 = M[-1][0], M[-1][1]
            v9, w9 = v8, w8 + ind_to_ground - wid/2

            # separate the inductor from the capacitor (OPTIONAL)

            if separated_elements==True:
                w9 = w9 - separation_length

            M.append([v9,w9])



    # GENERATE CAPACITOR AND INDUCTOR GEOMETRIES

    Upath = gdspy.FlexPath(U,t)    # capacitor
    Mpath = gdspy.FlexPath(M,wid)    # inductor

    # create the negative that will be used later to create the final resonator waveguide

    rec = gdspy.Rectangle((-((A/2) + dist_lateral_l), -add_t -dist_ground), ((A/2) + dist_lateral_r, B + ind_to_ground - wid))

    upneg = gdspy.boolean(rec, Upath, "not")
    upneg = gdspy.boolean(upneg, Mpath, "not")
    upneg = gdspy.boolean(upneg, bottom_cap, "not")

    # create the negative of a pair of resonators spaced by inter_cell_dist

    upneg_mirr = gdspy.copy(upneg)     # copy the geometry
    upneg_mirr.mirror([A/2 + dist_lateral_r + inter_cell_dist/2, 0],[A/2 + dist_lateral_r + inter_cell_dist/2, B])    # mirror the geometry
```

```python

    """2) FIRST RESONATOR"""


    # CHECK IF THERE ARE DIMENSIONS MISMATCHES

    while d_second < (y+w+(WG_width-w)/2):
        print("ERROR: dimensions mismatch, d_second smaller than y+w")
        new = input("Enter new value for d_second:")
        d_second = float(new)



    # INDUCTOR DESIGN

    #calculate horizontal dimension of inductor
    # this inductor horizontal dimension will define the distance
    ↪  between the capacitors's arm and the inductor

    bi = (3/5)*Ac - 2*tc

    #calculate number of windings in the inductor

    if compact_ind == False:
        N = int((Li-(d_second + (si+wi)))/(bi+si))
    else:
        N = round((Li-(d_second + (si+wi)))/(bi+si))

    #adapt start & end segment of inductor

    d_prime = (Li - (N*(si+bi) + d_second)) + wi

    # define set of points that will be used to draw the inductor

    M = []
    v0,w0 = centre[0], centre[1]
    M.append([v0,w0])
    v1,ww1 = v0, w0 + d_prime - wi/2
    M.append([v1,ww1])

    # Inductor's windings

    for i in range(N):

        if i%2 == 0:
            v2, ww2 = M[-1][0] + bi/2 - wi/2, M[-1][1]
            M.append([v2,ww2])
```

```
            v3, w3 = v2, ww2 + (si+wi)   # ww2 cause we already defined
             ↪  a w2 variable
            M.append([v3,w3])
            v4, w4 = v3 - (bi/2 - wi/2), w3
            M.append([v4,w4])

        if (i+1)%2==0:
            v5,w5 = M[-1][0] - bi/2 + wi/2, M[-1][1]
            M.append([v5,w5])
            v6, w6 = v5, w5 + (si+wi)
            M.append([v6,w6])
            v7,w7 = v6 + bi/2 - wi/2, w6
            M.append([v7,w7])

        if (i+1)==N:
            # print('end turns:', M[-1])
            v8,w8 = M[-1][0], M[-1][1]
            v9, w9 = v8, w8 + d_second + wi/2

            # separate the inductor from the capacitor (OPTIONAL)

            if separated_elements==True:
                w9 = w9 - separation_length

            M.append([v9,w9])

    # shift the inductor (shift every point) and center it with respect
    ↪  to the capacitor (you drew the inductor centred in (0,0))

    M = np.asarray(M)
    centring = (Ac/2)+Ws, tc
    M = M + centring

    # draw the inductor

    ind = gdspy.FlexPath(M,wi)    # FIRST RESONATOR INDUCTOR GEOMETRY




    # CAPACITOR DESIGN

    # Define length of the capacitor thin arm (the left one)

    Lt = N*(si+wi) + d_prime + tc

    # check for dimensions mismatch
```

```
    while (Lw >= (Lt+d_second)):
        print("MISMATCH: Lw is larger than Lt+d_second, it touches the ground")
        new = input("Enter new value for Lw:")
        Lw = float(new)

    while ((L2-w+sc) > (y+(Lt-tc))):
        print("ERROR: dimensions mismatch (L2 too large)")
        new = input("Enter new value for L2:")
        L2 = float(new)

    # define the capacitor geometry

    lcap = gdspy.Curve(0, 0).L(0,Lt, tc,Lt, tc,tc, Ws+Ac-tc,tc, Ws+Ac-tc,tc, Ws+Ac-tc,Lw, Ws+Ac+Wl-tc,Lw, Ws+Ac+Wl-tc,0, 0,0)
    cap = gdspy.Polygon(lcap.get_points()) # FIRST RESONATOR CAPACITOR GEOMETRY




    """2) FEEDLINE CAPACITOR DESIGN"""


    # DEFINE THE FEEDLINE CAPACITOR DESIGN

    # the center (0,0) is the same used for the first resonator design function

    wlcap = gdspy.Curve(-(sc+w1+Bwg),Lt+y).L(-(sc+w1+Bwg),Lt+y+w, tc+sc+w2,Lt+y+w, tc+sc+w2,Lt+y+w-L2, tc+sc,Lt+y+w-L2, tc+sc,Lt+y, -sc,Lt+y, -sc,Lt+y-L1, -(sc+w1),Lt+y-L1, -(sc+w1),Lt+y, -(sc+w1+Bwg),Lt+y)
    wcap = gdspy.Polygon(wlcap.get_points())


    # CREATE THE NEGATIVE of the whole resonator + waveguide capacitor geometry B

    contour_line = gdspy.Curve(-(sc+w1+first_dist_lateral_l),
```

```
    -dist_ground_first).L((Ws+Ac+Wl-tc+first_dist_lateral_r),(-dist_ground_first),(Ws+Ac+Wl-tc+first_dist_lateral_r),(Lt+d_second),-(sc+w1+first_dist_lateral_l),(Lt+d_second),-(sc+w1+first_dist_lateral_l),(Lt+d_second-(d_second-(y+w+((WG_width/2) - (w/2))))),-(sc+w1+Bwg),(Lt+d_second-(d_second-(y+w+((WG_width/2) - (w/2))))),-(sc+w1+Bwg),(Lt+d_second-(d_second-(y+w+((WG_width/2) - (w/2))))-WG_width),-(sc+w1+first_dist_lateral_l),(Lt+d_second-(d_second-(y+w+((WG_width/2) - (w/2))))-WG_width),-(sc+w1+first_dist_lateral_l),-(dist_ground_first))
    contour = gdspy.Polygon(contour_line.get_points())

    # boolean subtarction to generate the negative

    cont = gdspy.boolean(contour, wcap, "not")

    co = gdspy.boolean(cont, cap, "not")

    first_neg = gdspy.boolean(co, ind, "not") # NEGATIVE OF THE CELL CONTAINING FIRST RESONATOR + FEEDLINE CAPACITOR





    """3) LAST RESONATOR: mirrored copy of the first one"""


    # CREATE A MIRRORED COPY OF THE FIRST RESONATOR

    first_neg_mirr = gdspy.copy(first_neg)     # copy the geometry
    first_neg_mirr.mirror([(2*Lateral_r + N_pairs*Horizonta + (N_pairs-1)*inter_cell_dist)/2, 0],[(2*Lateral_r + N_pairs*Horizonta + (N_pairs-1)*inter_cell_dist)/2, 1*1e-6])    # mirror the geometry

    # NOTE: the mirrored copy is already shifted to the right at the end of the waveguide





    """4) GROUND DESIGN"""

```

```
    # CREATE THE GROUND RECTANGULAR GEOMETRY

    # calculate the ground rectangle length

    Rec_len = 2*Lateral_l + 2*Lateral_r + N_pairs*Horizonta +
    ↪  (N_pairs-1)*inter_cell_dist

    #create the ground rectangle

    ground = gdspy.Rectangle((-(Lateral_l),-(Height)), (Rec_len -
    ↪  Lateral_l, B + Height))


    # SUBTRACT THE NEGATIVES

    # create a cell array with the negatives of the standard resonators
    ↪  pairs

    Res_pair_cell = gdspy.Cell("std_resonators_pair_reference")

    Res_pair_cell.add(upneg)

    Res_pair_cell.add(upneg_mirr)

    res_array = gdspy.CellArray(ref_cell=Res_pair_cell, columns=N_pairs,
    ↪  rows=1, spacing = (Horizonta + inter_cell_dist, 0),
    ↪  origin=(Lateral_r + (half_res), 0))  # res_array type is cell
    ↪  array


    # added capacitor thickness for ANSYS and Sonnet simulations
    # check if there is a dimension mismatch

    while ((T1+T2) >= dist_ground):
        print("MISMATCH: ")
        new = input("Enter new value for T1:")
        T1 = float(new)
        new = input("Enter new value for T2:")
        T2 = float(new)


    rec1 = gdspy.Rectangle((Lateral_r + dist_lateral_l,0),(Lateral_r +
    ↪  dist_lateral_l + A,-T1))

```

```
    rec2 = gdspy.Rectangle((Lateral_r + dist_lateral_l,-dist_ground),(Lateral_r + dist_lateral_l + A, -dist_ground + T2)) # this has this weird geometry cause I want to maximize the distance between the the first resonator capacitor and the ground

    rec3 = gdspy.Rectangle((Lateral_r + Horizonta*N_pairs + inter_cell_dist*(N_pairs-1) - (dist_lateral_l + A) ,0),(Lateral_r + Horizonta*N_pairs + inter_cell_dist*(N_pairs-1) - (dist_lateral_l + A) + A,-T1))

    rec4 = gdspy.Rectangle((Lateral_r + Horizonta*N_pairs + inter_cell_dist*(N_pairs-1) - (dist_lateral_l + A),-dist_ground),(Lateral_r + Horizonta*N_pairs + inter_cell_dist*(N_pairs-1) - (dist_lateral_l + A)  + A, -dist_ground + T2))


    res_array = gdspy.boolean(res_array, rec1, "not")
    res_array = gdspy.boolean(res_array, rec2, "not")
    res_array = gdspy.boolean(res_array, rec3, "not")
    res_array = gdspy.boolean(res_array, rec4, "not")


    # add the negatives of the components to one single cell (called "negatives")

    negatives = gdspy.Cell("negatives")

    negatives.add(res_array)

    negatives.add(first_neg)

    negatives.add(first_neg_mirr)


    # subtract the negatives

    WG = gdspy.boolean(ground, negatives, "not")

    main = gdspy.Cell("Full_WG")
    main.add(WG)


    # RETURN: cell with the complete WG

    return main
```

```
#%%# ------------------------------------------------

# PRINT A GDS FILE OF THE CHOSED DESIGN

main = RES_WG_APO()

lib = gdspy.GdsLibrary()

lib.add(main)

gdspy.LayoutViewer(lib)

# Save the library in a file called 'first.gds'

lib.write_gds('file_name.gds')
```