DEVELOPMENT OF TUNABLE COMBLINE FILTER FOR SOFTWARE DEFINED RADIO (SDR) APPLICATIONS

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF TECHNOLOGY

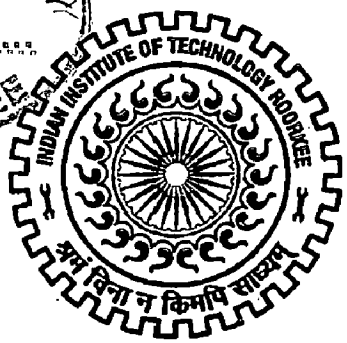
in

ELECTRONICS AND COMMUNICATION ENGINEERING

(With Specialization in RF & Microwave Engineering)

By

MADA.V.V.N.S. SANDEEP



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)

JUNE, 2010

CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this dissertation titled, **"Development of Tunable combline filter for Software Defined Radio (SDR) Applications"** in the partial fulfillment of the requirements for the award of degree of **Master of Technology** in **Electronics and Communication Engineering** with specialization in **Radio Frequency and Microwaves**, submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee, is an authentic record of my own work carried out from May 2009 to June 2010, under the supervision and guidance of **Dr. N. P. Pathak**, Assistant Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee.

I have not submitted the matter embodied in this dissertation for the award of any other Degree or Diploma.

Date: 29|06|10

Place: Roorkee

M Sandeep
29|06|10

(Mada.v.v.n.s.sandeep)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date: 29|06|10

Place: Roorkee

NPPathak
29|6|10

(Dr. N.P. Pathak)

Assistant Professor,
E&CE Department,
Indian Institute of Technology Roorkee,
Roorkee – 247 667

Acknowledgement

First and foremost I would like to thank my family for their constant love and support. I would not be the person I am without their influence and encouragement.

It gives me a great pleasure to take this opportunity and express my deep sense of gratitude to my supervisor, **Dr. N. P. Pathak**, Assistant Professor, Department of Electronics and Computer Engineering, for his valuable guidance and support. I am greatly in debt to him for providing the research facility required for the dissertation work. By working under him I had an opportunity to learn many things that helped me in nurturing my skills and understanding the intricacies of the dissertation.

I am grateful **to Prof. S. N Sinha**, Head of the Department of Electronics and Computer Engineering for creating the right infrastructure and facilities conductive to my dissertation work. I would like to thank the staff of Advance Microwave Laboratory and Wireless Communication Laboratory **Mr. Raja Ram, Mr. S.K.Gaur and Mr.Giri** for their valuable help and support.

I am greatly indebted to all my friends, who have graciously applied themselves to the task of helping me with ample morale support and valuable suggestions. Finally, I would like to extend my gratitude to all those persons who directly or indirectly helped me in the process and contributed towards this work.

(Mada.v.v.n.s.sandeep)

ABSTRACT

This thesis deals with the development of tunable Combline filter for Software defined radio (SDR) applications. The methodology for the design of Tunable Combline filter is explained in detail and explicit design formulas, to obtain the filter design parameters from specifications, are included. This tunable filter uses the concept of reverse biased diode as a Variable Capacitor (Varactor) for tuning purpose. The proposed filter is designed with the frequency range from 0.5GHz to 3GHz with commercial available varactors as tuning elements. The parasitic effects and simulation problems are discussed.

CONTENTS

List of Figures

List of Tables

Chapter -1
Literature Review

1.1. Introduction

The growth of mobile communications requires the design of miniaturized, multi-standard/multi-band, low cost transceivers. These need various tunable or reconfigurable components including a filter for Software Defined Radio (SDR) systems or multi band systems [1]. Software Defined Radio (SDR) is a radio communication technology that is based on software defined wireless communication protocols instead of hardwired implementations. In other words, frequency band, air interface protocol and functionality can be upgraded with software download and update instead of a complete hardware replacement. SDR provides an efficient and secure solution to the problem of building multi-mode, multi-band and multifunctional wireless communication devices [2].

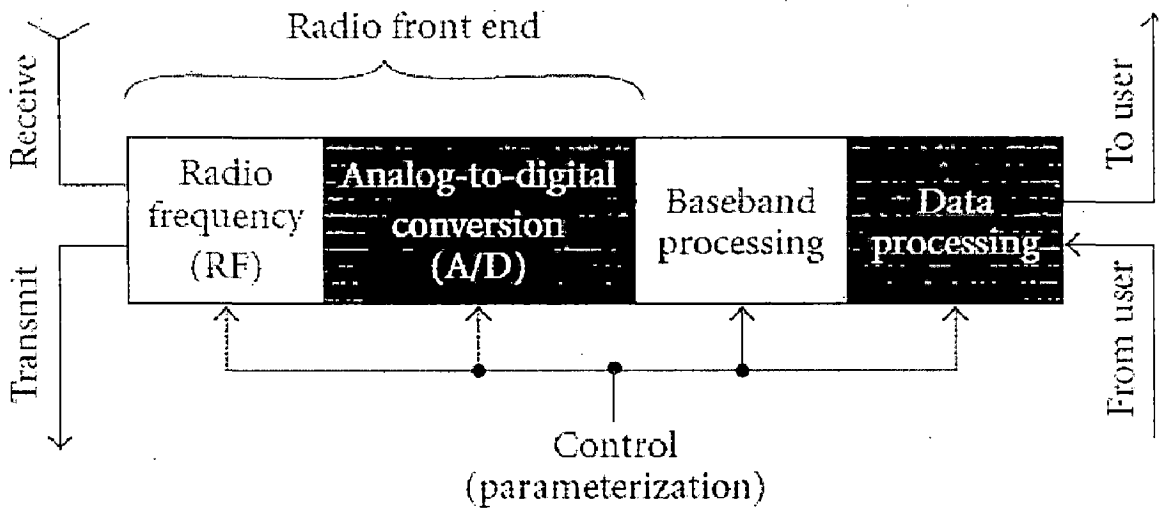


Fig 1.1 Software Defined Radio (SDR) Transceiver [2]

The fast-growing wireless market with the emergence of new wireless standards has created an increasing demand for multi-band and multi-standard base-station/mobile-handsets products with increased functionality and performance, while still meeting the requirements of smaller, lighter and cheaper.

These new standards, which set these requirements, involve more complex designs of RF transceiver circuits, where many passive components are intensively used for building of active devices such as amplifiers, oscillators, as well as for building of passive devices like filters. [2]

Filters are required to perform different functions like impedance matching or RF signal selection. Examples of filters in RF front-ends are pre-select filters, and image-reject filters. Image-reject filters are responsible for the rejection of the image signals, which are unwanted signals generated by the mixer and other components. Pre-select filters are needed for the selection of desired frequency bands while at the same time elimination of any undesired signals that may be present at the antenna's output at much higher power levels. These pre-select filters are often realized as duplexer, which serve to separate transmit signals from receive signals, which lie in an adjacent frequency band. Without these duplexers, the strong transmit signal would leak into the receiver section and saturate the low-noise-amplifier (LNA), making therefore the receiver to lose sensitivity to the weak receive signal. Generally, most of these passive components are realized as discrete components and therefore occupy more space.

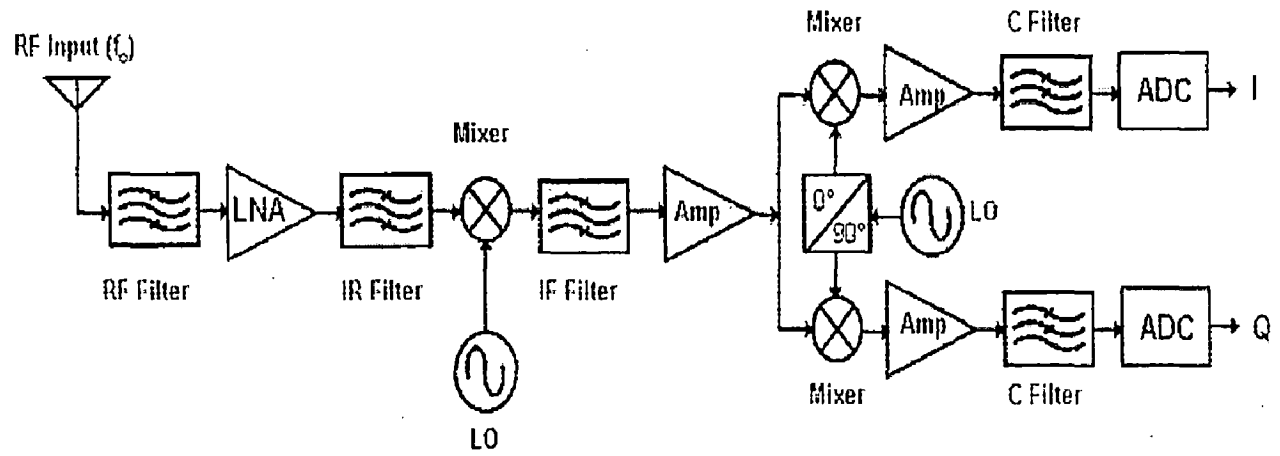


Fig1.2: Block diagram of Heterodyne Receiver architecture [3]

Heterodyne receivers (also called super heterodyne) are widely utilized for current wireless applications. Historically, heterodyne was the first practical receiver architecture implemented for cellular phone systems. A block diagram of a heterodyne receiver is shown in Fig. In this architecture, the signal received at the antenna is first filtered before being amplified by a low-noise amplifier (LNA) (e.g. by 10 to 20 dB). The signal is then further filtered by an image-reject (IR) filter before being frequency translated to an intermediate frequency (IF) by the first local oscillator (LO). At the IF stage the signal is further filtered by an IF filter and amplified before being frequency translated to base-band. Two parallel signal paths, in-phase (I) and quadrature (Q) are obtained at base-band after two-step frequency translation. Finally the I and Q signals are further amplified and filtered by low-pass filters,

respectively, before being converted to digital signals by the analog-to-digital converter (ADC).

1.2. Functions of filters in Wireless circuits

Filters are very important components of telecommunication systems. They are required to perform different functions like impedance matching or signal selection. Although similarities exist between impedance matching circuits and filter circuits just by looking at their schematics, there are distinct differences between these functions

Filters for impedance Matching

In RF systems the optimum power transfer is one of the important design considerations. The Power transfer needs to be maximized from one system block to another. This is achieved by means of matching networks. Matching networks are some type of electrical interconnections that are required between each building block. They are realized using strictly reactive and lossless components in order to achieve power conservation and to achieve usable gain from the active device (transistors) at microwave frequencies.

Filters for signal selection

The functions of filters in wireless circuits are mostly to reject the unwanted signal frequencies, while permitting a good transmission of the wanted ones. Depending on the circuit requirements, these filters can be designed as lowpass, highpass, bandpass, or band stop.

For the optimal operation of transceivers the isolation of both the transmitter and receiver have to be very large (e.g. 120 dB).The isolation between both the chains can be done using a duplexer (filter consisting of two band pass circuits), or a switch. Switches are mechanical, electrical, or electronic devices that open or close circuits, complete or break an electrical path, or select paths or circuits. The losses of a switch are usually less than those of a duplexer.

Filters in a transmitter chain are needed to meet the output noise requirements of the transmitter, since in some cases the transmitter noises can leak into the receiver via the duplexer and destroy the receiver sensitivity. The technology of choice for filters preceding the power amplifier (PA) is SAW (Surface Acoustic Wave). For output duplexers, ceramic

Band pass filters are also used. Ceramic filters have lower insertion loss and thus have less effect on the transmitter efficiency.

RF preselect filters

The purpose of RF preselect filters in receivers is to select the desired frequency band to be received, and to eliminate any undesired signal. These filters are typically realized in the form of a diplexer or duplexer. Duplexers connect the antenna to the transmitter (Tx) and receiver (Rx) and provides isolation between the Rx and Tx chains. These filters protect the receiver from saturation by interfering signals at the antenna and determine the receiver selectivity. Without the duplexer, the strong transmitted signal would leak into the receiver section and saturate the low-noise-amplifier, causing the receiver to lose sensitivity to the weak receive signal. Diplexers connect the antenna to a dual band transceiver and allow one frequency band to pass between the antenna and transceiver (e.g., connection of a GSM800- and PCS1900- dual band transceiver with antenna).

Ceramic coupled-resonator filters are commonly used for front-end receiver filters or duplexers. Lower-cost discrete LC filters are also used, especially in half-duplex transceivers where strong interference from the system's own transmitter is not an issue, and thus lower out-of band rejection is acceptable.

Image Reject filters

Image-reject filters (IR Filter in Figure 2.8) placed after the LNA are used to protect the RF mixer from out-of band interferer signals as well as to reject the undesired signals, generated by the RF mixer and other components. Without image rejection filtering, any signal present at the image frequency will be down converted to the same intermediate frequency (IF) and will corrupt the desired signal.

The bandwidth of IR filters, centered at the carrier frequency (f_c), must be sufficiently wide to pass the modulation sidebands in the desired channel without distortion. The image frequency has to be suppressed by at least 10 dB in order to meet the system noise figure (NF). To meet the system requirements, RF receivers generally need about 65 dB of image rejection. SAW (Surface Acoustic Wave) filters are typically used to provide image rejection. Low-cost LC filters can also be used at the expense of lower rejection levels.

Intermediate Frequency (IF) filters

Intermediate frequency filters (IF Filter) are designed to receive the entire RF passband and to reject spurious frequencies and image frequencies in particular. These IF filters narrow in and select the desired channel from the entire pass band; they provide thereby additional selectivity to the receiver. These filters help to prevent out-of-channel noise and help to minimize the loss, thus improving the sensitivity of receivers. SAW filters are the components of choice for most IF filter applications. They provide the best out-of-band rejection at reasonable size and cost. Crystal and LC filters can also be used at the expense of lower performance.

Base Band channel Filters

In baseband sections of wireless transceivers, filters for signal selection are required, e.g. antialiasing filters located before the A/D converter (C Filter in fig) and reconstruction filters after the D/A converters. These kinds of filters are generally low pass and remove the unwanted channels appearing at higher baseband frequencies, after down conversion. These filters are typically realized on-chip using silicon- or Ga As-technologies, for example. As can be seen throughout this section, many filters are required in wireless transceiver circuits, and particularly in the analog RF section. Since many of these filters are discrete or surface mounted, integrating them will lead to a substantial reduction in board size as well as in weight. For a successful integration however, efficient design and analysis methods are required.

1.3. Tuning Mechanisms

The tuning mechanism in tunable filters can be classified into three major types:

1. Mechanical Tuning
2. Magnetic Tuning
3. Electronic Tuning

Mechanically tunable bandpass filters are realized using co-axial or waveguide resonators and have large power-handling capability and low insertion loss. These manually tuned filters have slow tuning speed and are often large and bulky for application in modern integrated systems

Magnetically tunable band pass filters have been used extensively in microwave Communication systems and have single-crystal Yttrium-Iron-Garnet (YIG) spheres. In their

1.4.1. Varactor Tuned Combline Filter

Fig 1.3 shows a typical varactor tuned combline bandpass filter. The resonator consists of multi-microstrip transmission lines shorted at the same ends and loaded with varactor diodes at opposite ends. The transmission lines must have the electrical length which is less than that of a quarter wavelength line, in order to be resonating with the capacitance of varactor diode at the centre frequency of the filter. This configuration can provide a wide tuning range with minimum degradation of the pass band performance. A bandpass filter which has more than 0.45 octaves tuning range with less than 12.5% passband bandwidth variation has been reported.

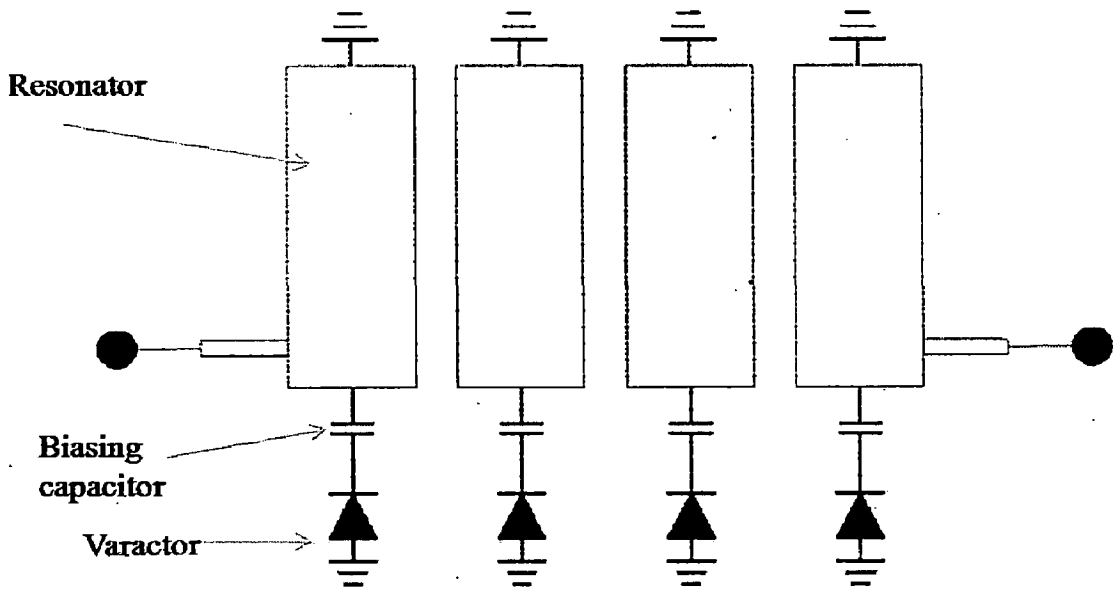


Fig 1.3 Typical Varactor tuned combline bandpass filter [5]

1.4.2. Varactor Tuned Interdigital Filter

The interdigital configuration is as commonly used [] as that of the combline, because only short positions are different and the varactor diodes are loaded at the open ends as are in the combline configuration in Fig 1.4. This configuration enables us to achieve wide tuning ranges more than 60% of its centre frequency. The centre frequency of the filter is also determined by the resonance length of the line as well as the capacitance value of the varactor diodes which tune the resonance frequency. The quality factor of the resonator is a function of the line length as well as the series resistance of the varactor diode.

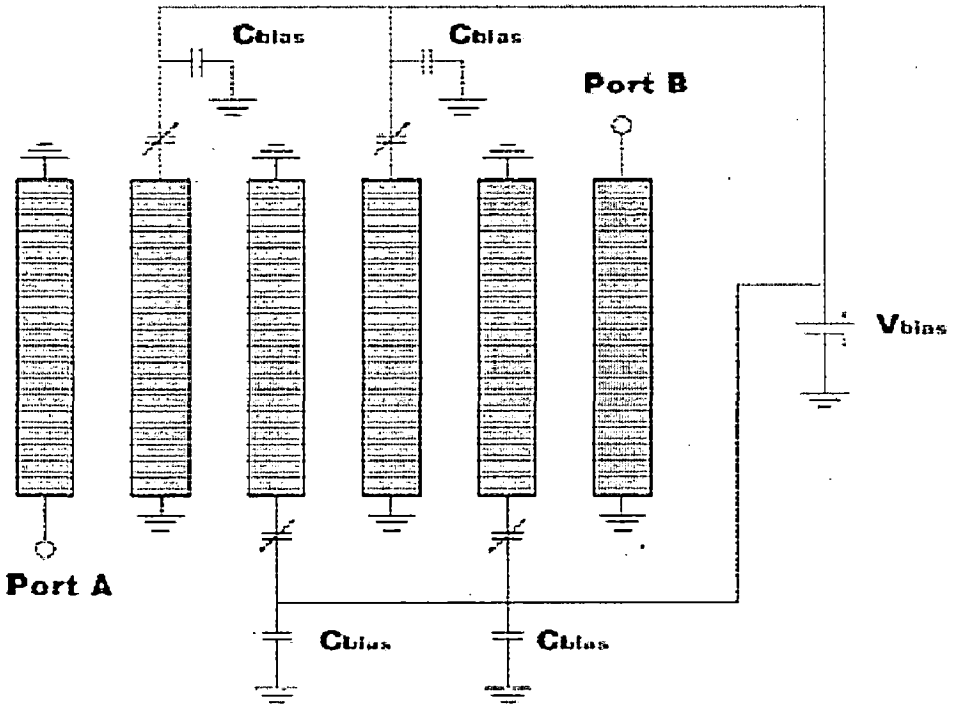


Fig 1.4 The varactor tuned interdigital bandpass filter [5]

1.4.3. Varactor Tuned Microstrip Line Ring Resonator

M.Makimoto proposed a tunable bandpass filter using microstrip ring resonators. The resonator is composed of transmission line and varactor diode located between the ends of the line shown in fig 1.5. This configuration gives relatively wide tuning range and steeper skirt frequency characteristics due to transmission zeros introduced by series resonance near the passband. The bandwidth of the second order filter is controlled by tight coupling characteristics between the two rings shown in fig 1.5.

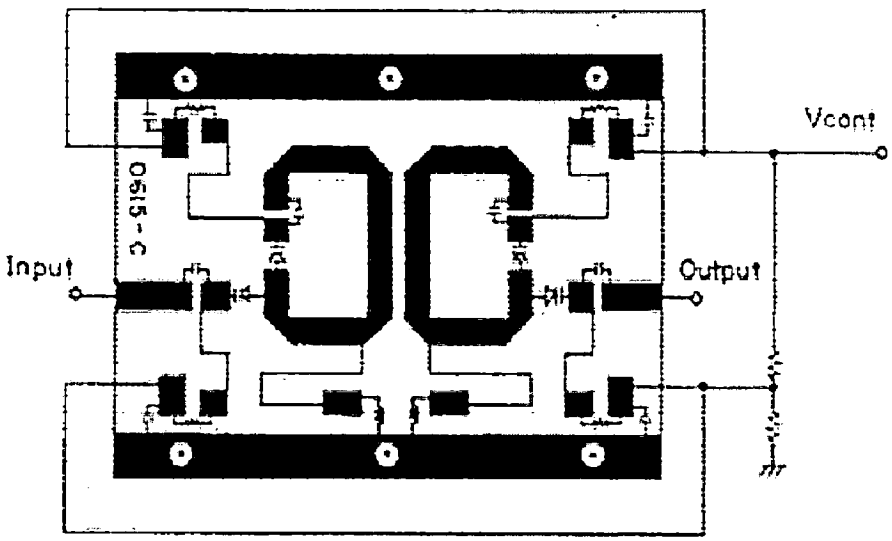


Fig 1.5 Typical Varactor tuned second order filter using microstrip ring resonator [5]

1.4.4. Varactor Tuned Hairpin Filter

Since the hairpin filter is more compact than the other configurations, it can be applied to miniature systems. The tunable bandpass filter has an octave band tuning range together with a compact size. Fig 1.6 shows a hairpin tunable bandpass filter which is composed of a half wavelength microstrip resonator and a tapped open stub. T-junction with a tapped open stub functions as equivalent K-inverter, and the bandwidth of the filter is dependent on the transmission line length and varactor (C_2) of that. The center frequency of the filter is also dependent on the resonator loaded varactor (C_1). Therefore, the center frequency as well as the bandwidth of the filter can be tuned. This configuration does not have wideband performances due to fixed input coupling capacitance.

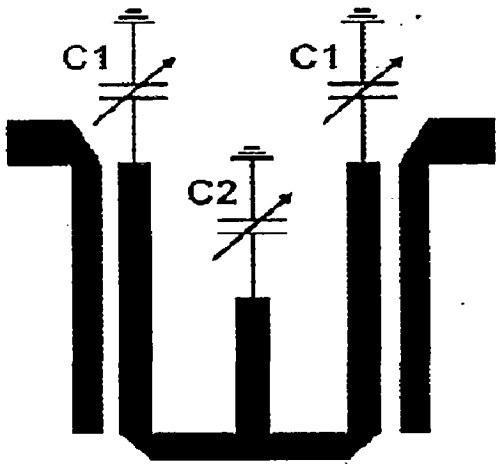


Fig 1.6 Second order varactor tuned hairpin filter [5]

1.5. Problem Definition

The work that is presented in this dissertation is aimed to design and develop a Electronically tunable combline band pass filter with varactors as tuning elements. The frequency range of operation of the filter is 0.5GHz to 3GHz.

1.6. Organization of the dissertation

Chapter 1 Provides literature review of the filters. It provides brief background of importance of filters in Software Defined Radio (SDR) and other wireless communication systems. Different methods of tuning mechanisms and different types of Varactor tuned bandpass filters are discussed, Problem statement, objective and scope of the thesis.

Chapter 2. Deals with the theory of filter design using insertion loss method, concept of resonant circuits, types of coupling between the resonant circuits, concept of admittance (J) and Impedance (K) inverters. This chapter concludes with the theory of tunable combline filter

Chapter 3 This chapter deals with the theory of electronically tunable combline filter with tunable centre frequency and variable bandwidth within the passband tuning range. This chapter provides a step by step procedure to design a tunable combline band pass filter starting from the specifications, Design parameters, transformation of electrical to physical dimensions of the filter is discussed and finally the designed filter is simulated by using ideal tunable capacitors using EM simulator ADS (Advanced Design Systems). Then the parasitic effects of the commercial varactors are discussed and the filter is simulated again by including these effects. Finally the filter is fabricated and tested.

Chapter 4 deals with the CAD tool for the design of tunable combline filter with tunable centre frequency and variable bandwidth within the pass band tuning range.

Chapter 5 draws conclusions, and finally describes the possible future research directions in the design of the tunable combline filter. This chapter deals how to control the bandwidth of the filter within the tuning range of the filter and how to compensate the parasitic effects of the varactors. Finally this chapter concludes with the concept of tuning using Ferroelectric materials like BST (Barium Strontium Titanate), $BaTiO_3$ (Barium titanate) for faster tuning

Chapter -2
Theory of Filter Design

2.1. Introduction

This chapter deals with the theory of tunable filter design, different types of filters , resonant circuits and the different methods of coupling between the resonant circuits, concept of admittance (J) inverters and impedance (K) inverters finally concludes with the tunable filtering mechanisms.

2.2 Fundamentals of Filter Design using Insertion loss method

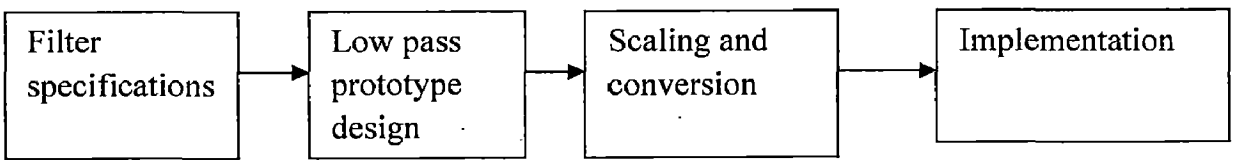
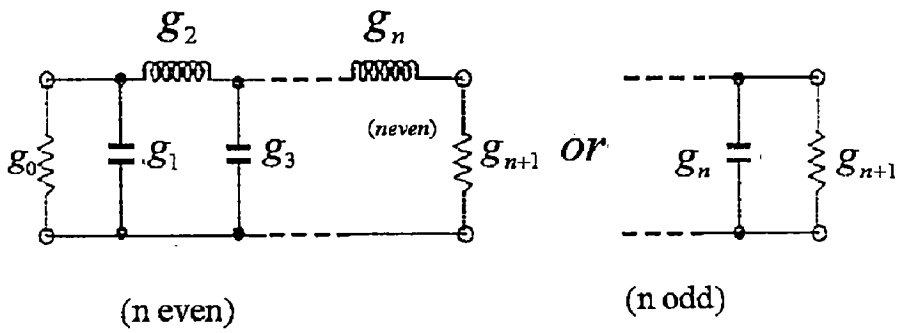
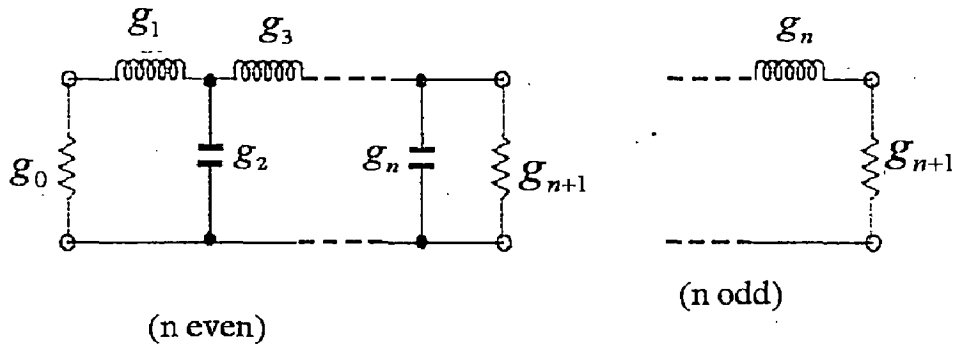


Fig 2.1: The process of filter design using insertion loss method [6]

Generally a low pass filter prototype is in general defined as the low pass filter, whose element values are normalized to make the source resistance or conductance equal to one, denoted by $g_0 = 1$, and the cutoff angular frequency to be unity, denoted by $\Omega_c = 1$(rad/s)



(a)



(b)

Fig 2.2 Low pass prototype filters with ladder network structure and its dual [7]

2.2.1 Prototype filter [8]

Butterworth low pass prototype filter:

For Butter worth response

$$L_A(\omega^1) = 10\log_{10}(1 + \left(\frac{\omega^c}{\omega_c^1}\right)^{2n}) \text{ dB} \qquad (2.1)$$

The g notation used in figure signifies roots of an nth order transfer function that governs its characteristics. These represents the normalized values of filter elements with a cut off frequency of $\Omega_c = 1$(rad/s).

Element values for normalized Butterworth low pass prototype filter are:

$g_0 = 1$

$$g_k = 2*\sin\left(\frac{(2k-1)\pi}{2n}\right), \quad k = 1, 2 \ldots . n \qquad (2.2)$$

$g_{n+1} = 1$

To determine the order of a Butterworth low pass prototype filter, the specification that usually given is the stop band attenuation (L_{AS} in dB) at $\Omega = \Omega_s$ for $\Omega_s > 1$.

$$n > \frac{\log(10^{\frac{L_{AS}}{10}} - 1)}{2\log\Omega_S} \qquad (2.3)$$

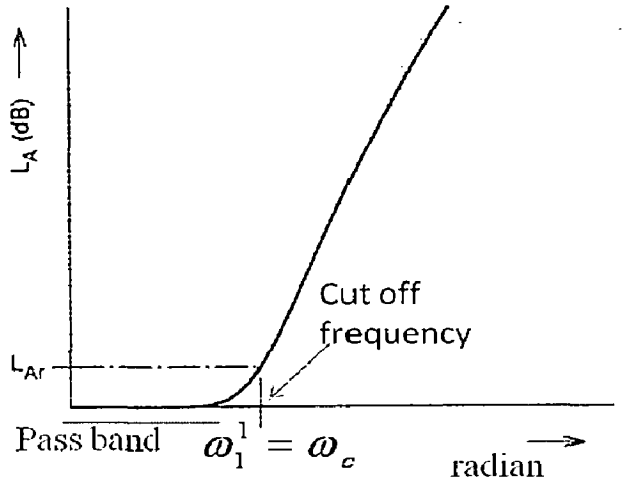


Fig 2.3: Butterworth (Maximally flat) response [7]

Since quantity in square bracket in eqn has 2n zero derivatives at ω^1=0, hence its name is "maximally flat".

Chebyshev Low Pass Prototype filters:

For chebyshev low pass prototype filters having a transfer function given in

$$\text{ripple constant} = \varepsilon = \sqrt{10^{\frac{L_{Ar}}{10}} - 1} \tag{2.4}$$

Where L_{Ar} is the pass band ripple in dB.

The element values of the two port network of may be computed using the following formulas:

$g_0 = 1.0$

$$g_1 = \frac{2}{\gamma}\sin\left(\frac{\pi}{2n}\right) \tag{2.5}$$

$$g_i = \frac{1}{g_{i-1}} \frac{4\sin\left(\frac{(2i-1)\pi}{2n}\right)\times \sin\left(\frac{(2i-3)\pi}{2n}\right)}{\gamma^2 + \sin^2\left(\frac{(i-1)\pi}{n}\right)} \quad \text{For i} = 2, 3\ldots n \tag{2.6}$$

$$g_{n+1} = 1.0 \qquad \text{For n odd}$$

$$\coth^2\left(\frac{\beta}{4}\right) \qquad \text{For n even} \tag{2.7}$$

$$\beta = \ln[\coth\left(\frac{L_{Ar}}{17.37}\right)] \tag{2.8}$$

$$\gamma = \sinh(\frac{\beta}{2n}) \tag{2.9}$$

for the required pass band ripple ,minimum stop band attenuation the order of the chebyshev lowpass filter is found by

$$n \geq \frac{\cosh^{-1}\sqrt{\frac{10^{0.1L_{As}}-1}{10^{0.1L_{Ar}}-1}}}{\cosh^{-1}\Omega_s} \tag{2.10}$$

Where L_{As} is the stop band attenuation at Ω_s

For 'n' odd, we have equal source and load impedances, but for 'n' even we have unequal source and load impedances. If design leads to 'n'=even and we need equal source and load impedances increase either 'n' by one or put a impedance matching network at the load end.

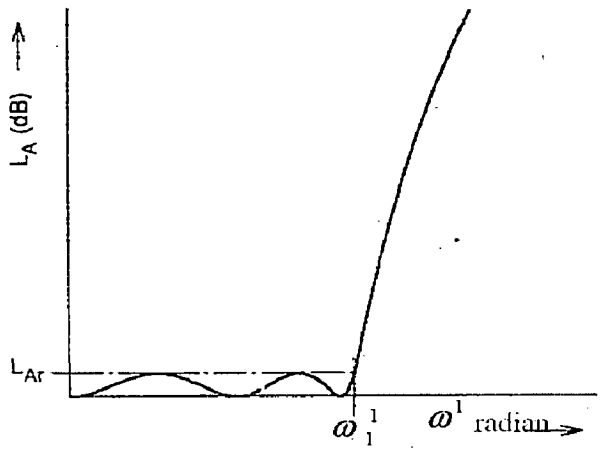


Fig 2.4 : Chebyshev attenuation characteristics [7]

L_{Ar} Is the maximum attenuation (dB) in the passband

ω_1^1 = Equal ripple band edge

For $\omega^1 < \omega_1^1$

$$L_A(\omega^1) = 10\log_{10}(1 + \epsilon\cos^2(n\cos^{-1}(\frac{\omega^1}{\omega_1^1}))) \tag{2.11}$$

and $\omega^1 > \omega_1^1$

$$L_A(\omega^1) = 10\log_{10}(1 + \epsilon\cosh^2(n\cos^{-1}(\frac{\omega^1}{\omega_1^1}))) \tag{2.12}$$

For both maximally flat as well as chebyshev type response, "n" is the order or number of reactive elements present in the circuit.

For a lowpass chebyshev response, if n=even, there are $(\frac{n}{2})$ frequencies where L_A=0 while if n= odd there will be $(\frac{n+1}{2})$ where L_A= 0

2.2.2. Impedance and Frequency Scaling [6]

Impedance scaling:

In the prototype design, the source and load resistances are unity .A source resistance of R_0 can be obtained by multiplying the impedances of the prototype design by R_0 .Let prime denotes impedance scaled quantities. New filter component values are given by

$$L^1=R_0L \tag{2.13}$$

$$C^1=\frac{C}{R_0} \tag{2.14}$$

$$R_s^1=R_0 \tag{2.15}$$

$$R_L^1=R_0R_L \tag{2.16}$$

L, C and R_L are the component values for the original prototype filter.

Frequency scaling:

For changing the cutoff frequency of a low pass prototype from unity to Ω_c requires the scaling of frequency by the factor of $\frac{1}{\omega_c}$ which is accomplished by replacing ω by $\frac{\omega}{\omega_c}$.

Thus new elements values are obtained by applying the substitution of $\omega \longrightarrow \frac{\omega}{\omega_c}$ to the series reactances and shunt susceptance of the prototype filter.

Thus

$$jX_k=j\frac{\omega}{\omega_c}L_k=j\omega L_k^1 \tag{2.17}$$

$$jB_k=j\frac{\omega}{\omega_c}C_k=j\omega C_k^1 \tag{2.18}$$

When both impedance and frequency scaling is done then the new element values then obtained are

$$L_k^1=\frac{R_0L_k}{\omega_c} \tag{2.19}$$

$$C_k^1 = \frac{C_k}{R_0 \omega_c} \qquad (2.20)$$

2.2.3. Filter Transformations:

Low pass to high pass transformation:

The frequency substitution where $\omega \longrightarrow \frac{-\omega_c}{\omega}$ can be used to convert a low pass response to a high pass response .so the basic elements of the low pass elements like inductor is converted to capacitor and capacitor is converted into inductor in high pass filter transformation.

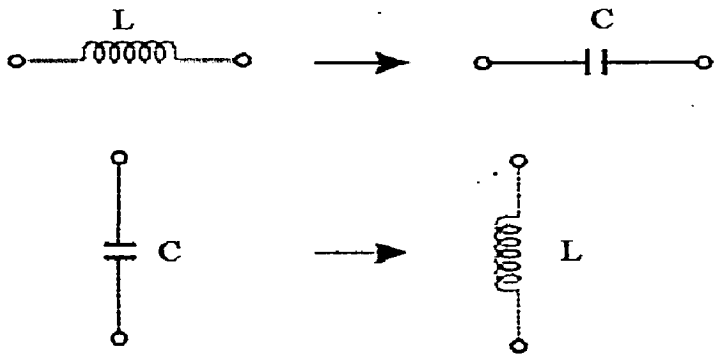


Fig 2.5 : Low pass to high pass transformation [7]

Low pass to band pass transformation:

The frequency substitution where $\omega \longrightarrow \frac{\omega_0}{\omega_2 - \omega_1}\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right) = \frac{1}{\Delta}\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right)$

Where Δ is fractional bandwidth of the pass band ω_1 and ω_2 denote the edges of the pass band and the centre frequency $\omega_0 = \sqrt{\omega_1 * \omega_2}$

In this transformation series inductor of low pass filter is converted to series LC circuit and shunt capacitor of low pass filter is converted to parallel LC circuit

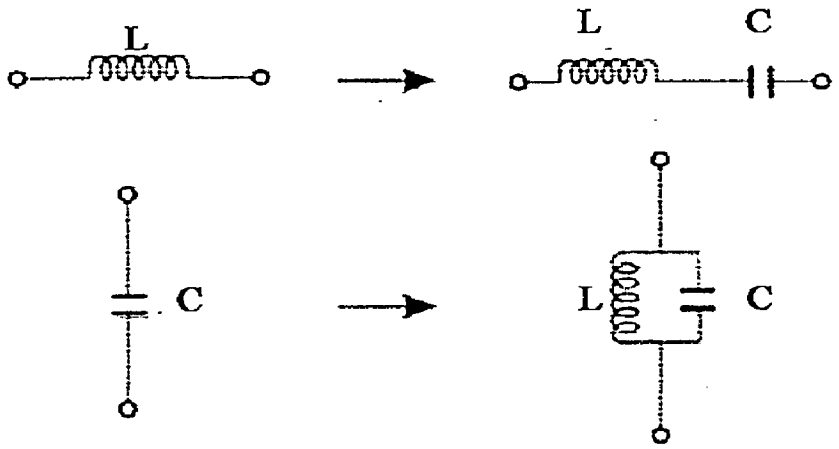


Fig 2.6 : Low pass to band pass transformation [7]

Low pass to band stop transformation:

The frequency substitution where $\omega \longrightarrow \Delta(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega})^{-1}$ is the general transformation for low pass to band stop transformation .In this transformation series inductor is replaced by parallel LC circuit and shunt C is replaced by series LC circuit

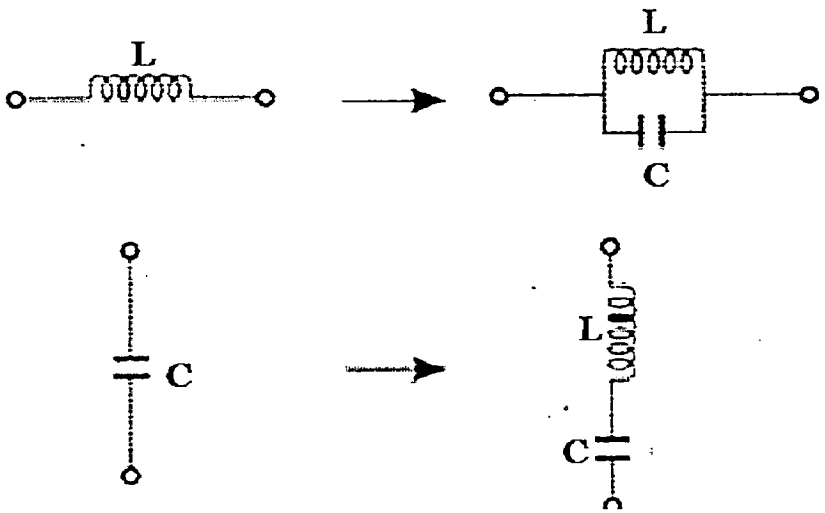


Fig 2.7: Low pass to band stop transformation [7]

2.3. Resonant Circuits

Resonant circuits are used in practically every transmitter , receiver to selectively pass a certain frequency or group of frequencies from a source to a load while attenuating all other frequencies outside of this passband.The perfect resonant circuit pass band would appear as

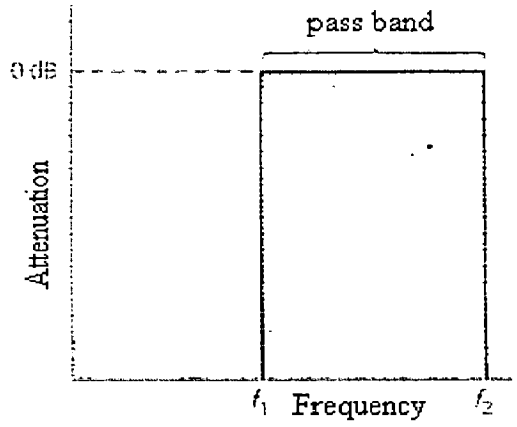


Fig 2.8 :The perfect filter response [9]

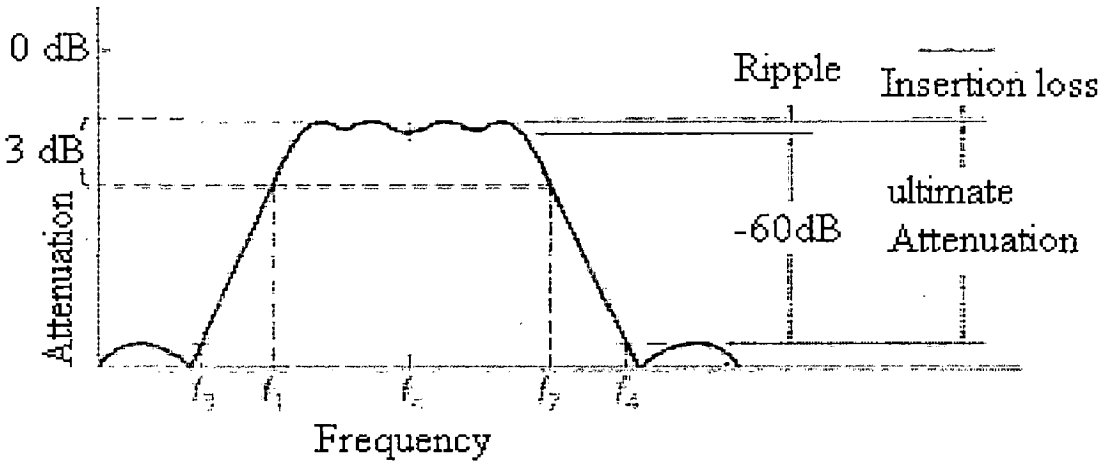


Fig 2.9: A practical filter response [9]

Quality Factor

The quality factor of a resonator (Q) is a figure of merit that determines the energy dissipated by a resonant circuit. The general definition for the Q-factor is given by,

$$Q = \omega \frac{\text{Energy stored in resonator}}{\text{Power dissipated}}$$

At Resonance Q- factor is

$$Q = \omega_0 \frac{W_m + W_e}{P_{loss}} = \omega_0 \frac{2W_m}{P_{loss}} \qquad (2.21)$$

Where ω_0 is the resonant frequency and W_m , W_e are the average Electric and Magnetic energies stored in the resonant structures and P_{loss} is the power loss. At the resonant frequency both the Magnetic and Electric energies are equal.

In practice, for a resonator to be used in a system, it has to be loaded by external Circuitry. The losses associated with the external circuitry add to the losses in the resonant structure and

affects the Q -factor measurement of the resonant structure. In Order to measure the Q-factor of the resonant structure exactly, the Q -factor associated with the external circuitry (Q_{ext}) has to be known. The Q-factor of the resonant Structure and the external circuitry is called the loaded Q-factor (Q_L). The Q-factor associated with the resonant structure alone is called the unloaded Q-factor (Q_u) of the structure, Q_{ext}, Q_L and Q_u are related as

$$\frac{1}{Q_L} = \frac{1}{Q_u} + \frac{1}{Q_{ext}} \tag{2.22}$$

The unloaded Q-factor for the end-coupled and tapped resonator filters are approximately 140-150. This reasonably high value of Q factor is a result of low loss in the filter and justifies the use of microstrip designs for filters in front-end receiver electronics. The loss can be further decreased by housing the filter in a metal casing such that radiation losses are minimized thus increasing unloaded Q-factor of the filter. This is especially important in microstrip circuits since they tend to have high radiation losses due to fringing fields at the edges of the microstrip line.

RF and Microwave Resonators

RF and microwave resonators are lumped element networks or distributed circuit structures that exhibit minimum or maximum real impedance at a single frequency or at multiple frequencies. The resonant frequency f_0 is the frequency at which the input impedance or admittance is real. The resonant frequency may be further defined in terms of series or shunt mode of resonance. The series mode is associated with small values of input resistance at the resonant frequency, while the shunt mode is associated with large values of resistance at the resonant frequency. Some typical lumped and distributed resonators are shown in

Resonators may be characterized by their unloaded quality factor Q_u which is the ratio of the energy stored to the energy dissipated per cycle of the resonant frequency .Resonators are also characterized with respect to their reactance (α) or susceptance (β) slope parameters,which are defined respectively as

$$\alpha = \frac{\omega_0}{2}\frac{dX(\omega)}{d\omega} \text{ at } \omega = \omega_0 \text{ and } \beta = \frac{\omega_0}{2}\frac{dB(\omega)}{d\omega} \text{ at } \omega = \omega_0$$

These are important resonator parameters because they influency Q_u and the coupling factor between resonators in multiple resonator filters. The following table provides the reactance

and susceptance slope parameters of some common lumped element and distributed resonators.

Resonator type	Reactance slope (α)	Susceptance slope (β)
Series LC	$\omega_0 L$ or $\frac{1}{\omega_0 C}$	
Shunt LC		$\omega_0 C$ or $\frac{1}{\omega_0 L}$
Shunt π		$2\omega_0 C$
$\frac{\lambda}{2}$ Line (short)	$\frac{\pi Z_0}{2}$	
$\frac{\lambda}{2}$ Line (open)		$\frac{\pi Y_0}{2}$
$\frac{\lambda}{8}$ Line (short + C)		$\frac{Y_0}{2}(\cot\theta_0 + \theta_0 \csc\theta_0^2)$
$\frac{\lambda}{4}$ Line (short)		$\frac{\pi Y_0}{4}$

Table 2.1: Slope Parameters for Different Types of Resonators [3]

Q_u= may also be defined in terms of the reactance or susceptance slope parameters as

$Q_u = \frac{\alpha}{R_{se}} = \beta R_{sh}$ where R_{se} = Resonator series resistance ,R_{sh} = Resonator shunt resistance.

Together these resistances represent the resonator loss.

2.4. Coupling of Resonant circuits

The coupling mechanism that is used is generally chosen specifically for each application,as each type of coupling has its own peculiarcharacteristics .The most common forms of coupling are :capcitive, inductive ,transformer(mutual) and active (transistor)

Capacitive Coupling

Capacitive coupling is probably the most frequently used method of linking two or more resonant circuits.

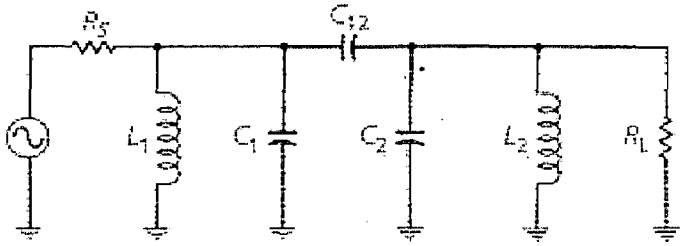


Fig 2.10: Circuit arrangement for a two resonator capacitively coupled filter [9]

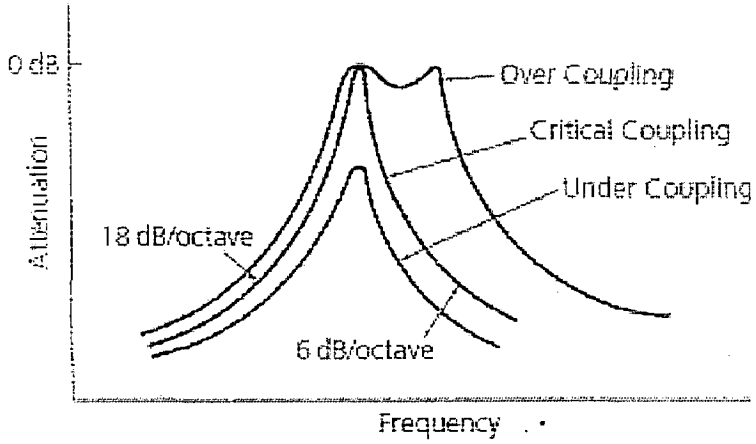


Fig 2.11: The effects of various values of capacitive coupling on pass band response [9]

Inductive Coupling

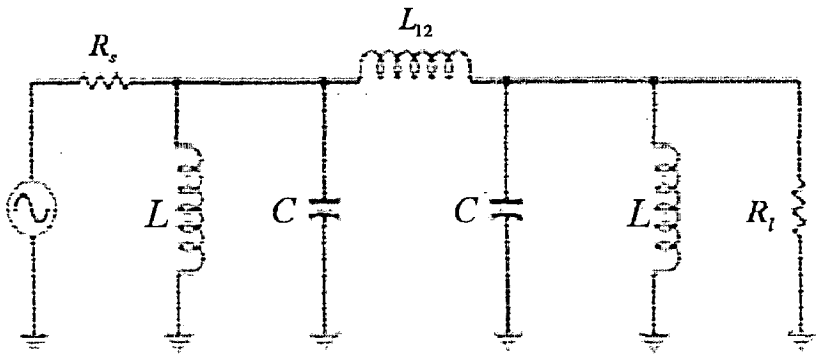


Fig 2.12: Circuit arrangement for a two resonator Inductive coupled filter [9]

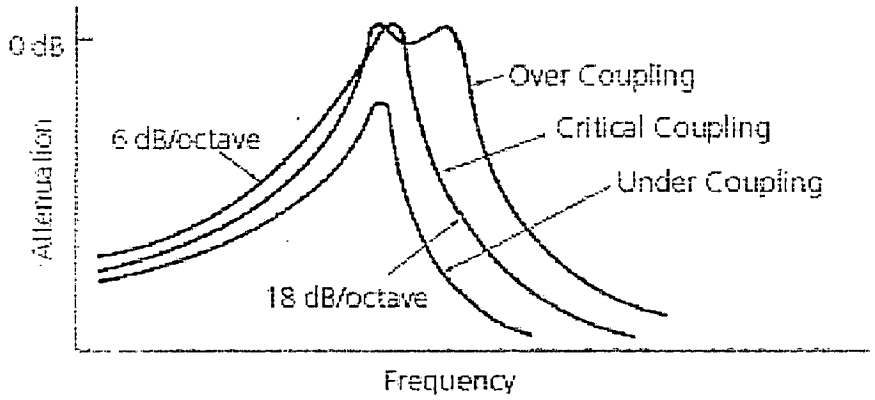


Fig 2.13: The effects of various values of inductive coupling on pass band response [9]

2.5. Immitance Inverters

Immitance inverters are either impedance or admittance inverters. An idealized impedance inverter is a two port network that has a unique property at all frequencies ,that means if it is terminated in a n impedance Z_2 on one port, the impedance Z_1 seen looking in at the other port is $Z_1 = \frac{K^2}{Z_2}$ where K is real and defined as the characteristic impedance of the inverter. As can be seen, if Z_2 is inductive/conductive, Z_1 will become conductive/inductive, and hence inverter has a phase shift of ± 90 degrees .Impedance inverters are also known as K-inverter. The ABCD matrix of ideal impedance inverters may generally be expressed as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 0 & jK \\ \frac{1}{jk} & 0 \end{bmatrix} \tag{2.23}$$

An ideal admittance inverter is a two-port network that exhibits such a property at all frequency that if an admittance Y_2 is connected at one port, the admittance Y_1 seen looking in the other port is $Y_1 = \frac{J^2}{Y_2}$ where J is real and called the characteristic admittance of the inverter. Similarly, the admittance inverter has a phase shift of ± 90 degrees .Admittance inverters are also referred to as J-inverters. In general, ideal admittance inverters have the ABCD matrix

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 0 & jJ \\ \frac{1}{jJ} & 0 \end{bmatrix} \tag{2.24}$$

A series inductance with an inverter on each side looks like a shunt capacitance from its exterior terminals, likewise, a shunt capacitance with an inverter on each inverter on each side looks like a series inductance from its external terminals.

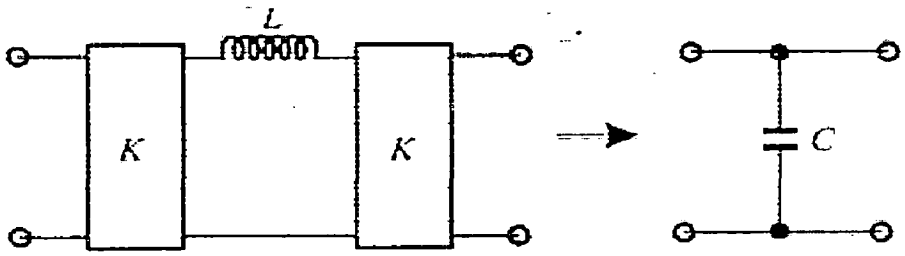


Fig 2.14: Impedance inverters used to convert a series inductor to shunt capacitor [7]

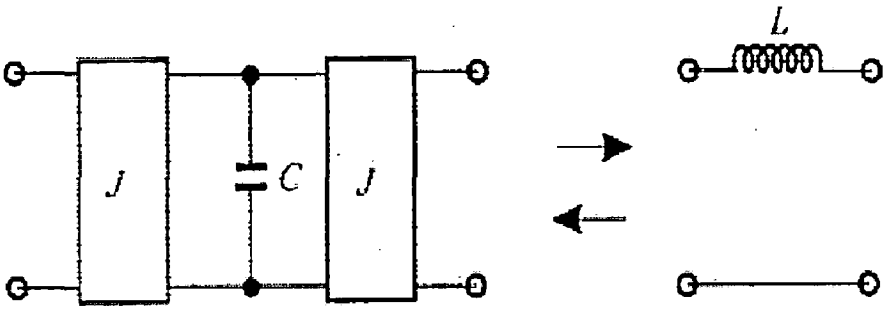


Fig 2.15: admittance inverter used to convert a shunt capacitor to series inductor [7]

Inverters have the ability to shift impedance or admittance levels depending on the choice of K or J parameters. Making use of these properties enables us to convert a filter circuit to an equivalent form that would be more convenient for implementation with microwave structures.

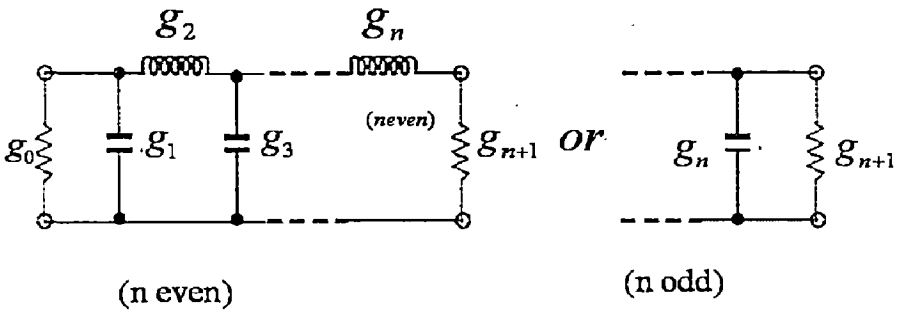


Fig 2.16: Low pass filter prototype [10]

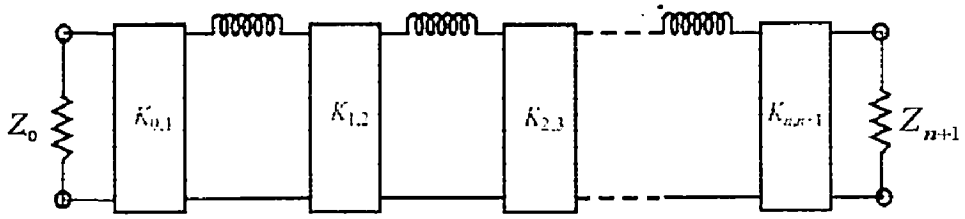
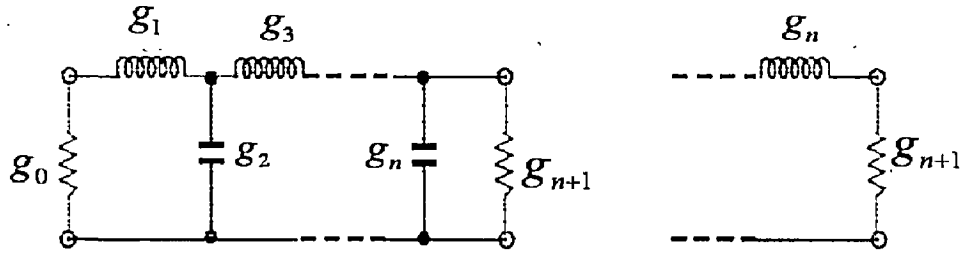


Fig 2.17: Low pass filter proto type modified with impedance inverter [10]



(n even)

(n odd)

Fig 2.18: Dual of Low pass prototype filter

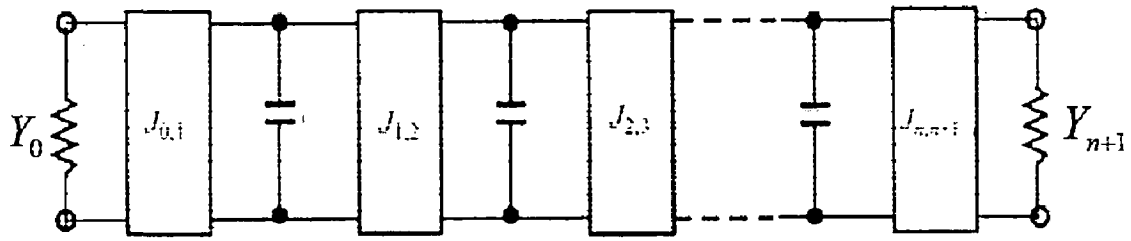


Fig 2.19: Low pass filter modified with Admittance inverter

The low pass prototype filter can easily be transformed to bandpass filter with shunt parallel resonators as shown in fig:

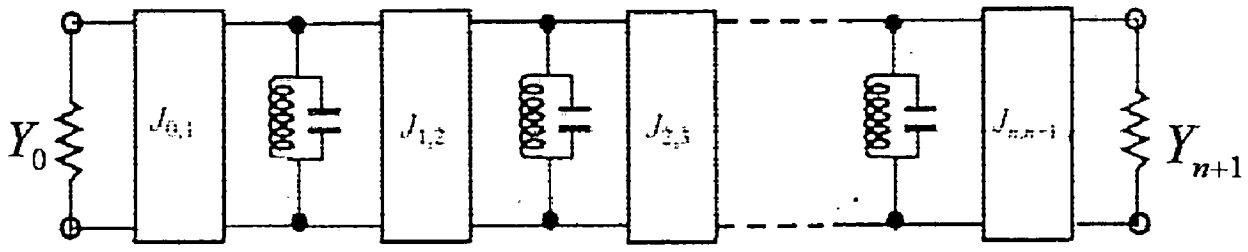


Fig 2.20: Band pass filter using Admittance inverter

Important generalizations are obtained by replacing the Lumped LC resonators by distributed circuits. Distributed circuits can be microstrip resonators, or any other suitable resonant structures. In the ideal case, the reactance or susceptances of the distributed circuits should be equal those of the lumped resonators only near resonance.

2.6. Coupled Lines

When two unshielded transmission lines are close together, power can be coupled between the lines due to the interaction of the electromagnetic fields of each lines. Such lines are referred to as coupled transmission lines .Coupled transmission lines are usually assumed to operate in TEM mode, which is rigorously valid for stripline structures and approximately valid for microstrip structures. Coupled line structures can be used to implement directional couplers, hybrids, and filters.

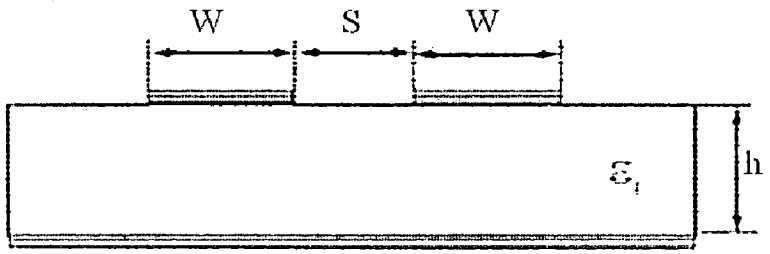


Fig 2.21: cross section of coupled microstrip lines [11]

Coupled Line Theory

The coupled lines or any other three wire line, can be represented by the structure shown in fig.. If we assume TEM mode of propogation, then the electrical characteristics of the coupled lines can be completely determined from the effective capacitances between the lines and the velocity of propogation on the line.

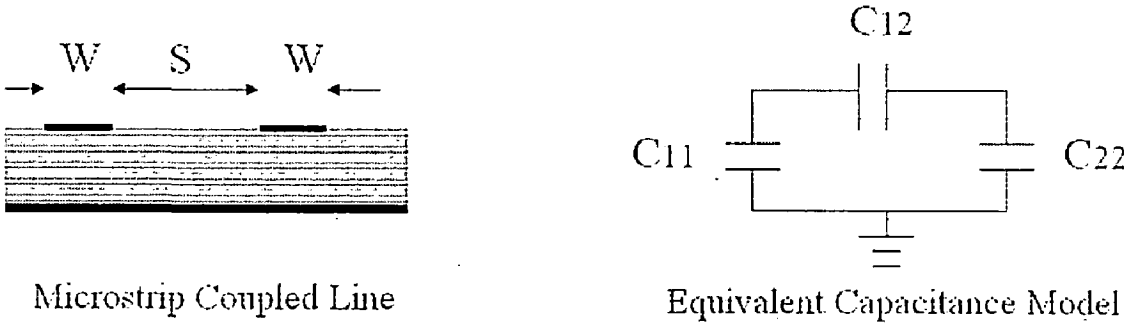


Fig 2.22: Microstrip coupled lines and its equivalent capacitor model

Types Of Excitations For The Coupled Line

1. Even mode of excitation

2. Odd mode of excitation

For the even mode of excitation the currents in the strip conductors are equal in amplitude and in the same direction, and the odd mode, where the currents in the strip conductors are equal in amplitude but in opposite direction.

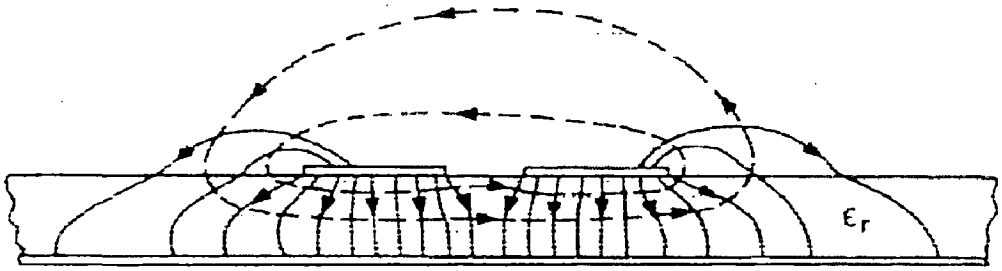


Fig 2.23: Electric and Magnetic Field lines of a coupled line operating in even mode

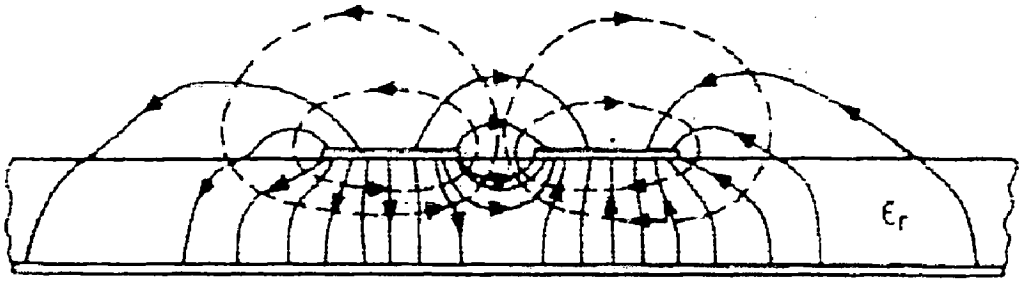


Fig 2.24: Electric and Magnetic Field lines of a coupled line operating in odd mode

For the even mode, the electric field has even symmetry about the center line, and no current flows between the two strip conductors. This leads to the equivalent circuit shown, where C_{12} is effectively open circuited. Then the resulting capacitance of either line to ground for the even mode is

$$C_e = C_{11} = C_{22}$$

Characteristic impedance for the even mode is

$Z_{0e} = \frac{1}{V_p C_e}$. Where V_p is the phase velocity of propogation on the line

For the odd mode, the electric field lines have an odd symmetry about the centre line, and a voltage null exists between the two strip conductors. We can imagine this as a ground plane through the middle of C_{12}. In this case, the effective capacitance between either strip conductor and ground is

$$C_0 = C_{11} + 2C_{12} = C_{22} + 2C_{12} \tag{2.25}$$

Characteristic impedance for the odd mode is

$Z_{0o} = \frac{1}{v_p C_0}$, Where V_p is the phase velocity of propogation on the line

An arbitrary excitation of a coupled line can always be treated as a superposition of approximate amplitudes of even and odd modes.

2.7. Theory of Tunable Combline filter

A Electronically tunable combline filter consists of coupled lines shorted on one end and terminated with varactor tuning diodes on the other. Combline structures are grounded on one end and are capacitively loaded at the open end. If the lines are a quarter wavelength long, magnetic coupling near the grounded ends and electrostatic coupling at the open ends are equal in magnitude but opposite in phase, resulting in no coupling (an all-stop structure).With capacitive loading at the open ends, the electrostatic coupling sections are shortened and coupling is predominantly magnetic.

Combline filters have the following attractive features over other conventional filters

(a) They are compact

(b) They have strong stop bands, and the stop bands above the primary pass band can be made to be strong

(c) If desired, they can be designed to have an unusually steep rate of cutoff on the high side of the pass band

(d) Adequate coupling can be maintained between resonator elements with sizeable spacings between resonator lines.

(e) Filters of this type can usually be fabricated without dielectric support materials so that ,if desired, dielectric losses can be eliminated.

In the case of a tunable band pass, the length of the combline is selected such that the coupling between resonators is reduced at the proper rate as the frequency is increased.. Although several methods exist for exciting the resonator, input and output tapping is one of the easiest. A typical tapped combline filter is shown in fig. 2.25.

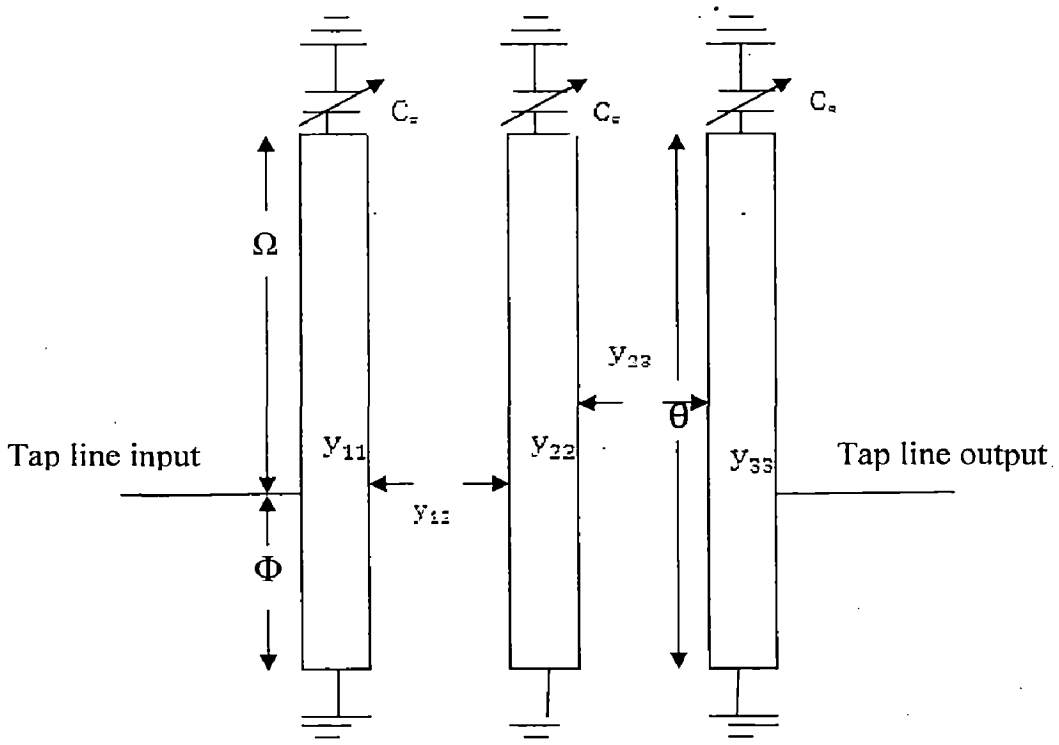


Fig 2.25: Geometry for tap line input output tunable combline filter

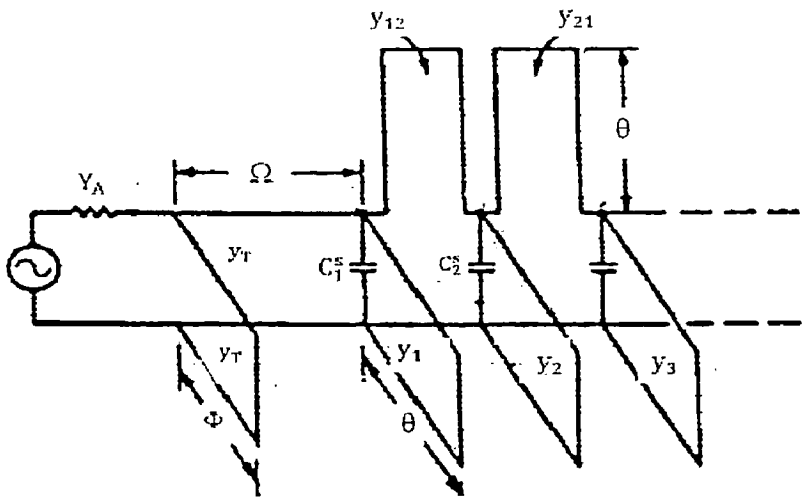


Fig 2.26: Transmission line Equivalent circuit for Tapped line input combline filter [12]

Where

$$y_T = y_{11} - \frac{y_{12}^2}{y_{22}}, \tag{2.26}$$

$$y_1 = \frac{y_{12}}{y_{22}}(y_{12} - y_{22}), \tag{2.27}$$

$$y_2 = y_{22} - y_{12} - y_{23}, \tag{2.28}$$

$$y_3 = y_{33} - y_{23} \tag{2.29}$$

all line lengths not indicated = θ

The equivalent circuit suggested by Cristal for the tapped line input combline filter is shown in Fig .The admittance inverter used to derive the equations for the combline filter consists of pi configuration, i.e., a series shorted stub of characteristic admittance of y_{12} and two shunt shorted stubs of characteristic admittance $-y_{12}$ .Therefore y_1 is split into two parts, with $y_L = \left(\frac{y_{12}^2}{y_{22}}\right)$ added on the left to the input stage and $y_R = -y_{12}$ forming the shunt stub of the admittance inverter J_{12} .The input stage of the filter now consists of four elements as

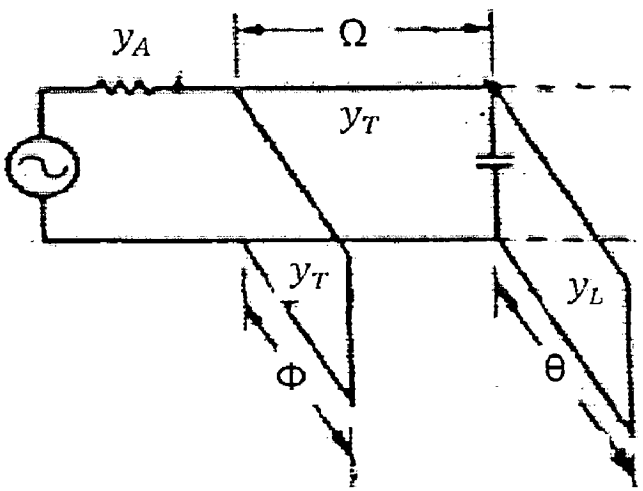


Fig 2.27: Circuit of Tapped line input stage of combline filter [12]

New Equivalent circuit for input stage: The ABCD matrix of the first two elements is

$$\begin{bmatrix} 1 & 0 \\ -jy_T \cot\emptyset & 1 \end{bmatrix} \begin{bmatrix} \cos\Omega & jZ_T \sin\Omega \\ jy_T \sin\Omega & \cos\Omega \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ \Gamma & \delta \end{bmatrix} \tag{2.30}$$

Where θ is the electrical length of resonators

$\emptyset$ is the electrical length from ground to input line

$\Omega = \theta - \emptyset$

y_T (Z_T) admittance (impedance) of the first and second elements

Multiplying the above matrix by the ABCD matrix of the other elements of the input stage (C_1^s , y_L), The ABCD matrix of the input stage becomes

$$\begin{bmatrix}\alpha & \beta\\ \Gamma & \delta\end{bmatrix}\begin{bmatrix}1 & 0\\ y & 1\end{bmatrix}=\begin{bmatrix}\alpha+\beta y & \beta\\ \Gamma+\delta y & \delta\end{bmatrix} \tag{2.31}$$

$$\text{Where}\quad y=j\,\omega C_1^s - jy_L \cot\theta \tag{2.32}$$

The series inductance must be compensated for by introducing an additional shunt capacitance C_c^s at the first resonator. This is the capacitor that found necessary to introduce when comparing the tapped input filter to the transformer input filter.

At the resonant frequency of the filter, the lines are significantly less than a quarter wavelength long, thus the filter is physically compact and it possesses a broad stop band bandwidth.

The following figure shows the equivalent circuit of the combline filter. The resonators are composed of distributed inductors in parallel with lumped capacitors, while the coupling between the resonators is via series distributed inductors.

If the series coupling inductors are now shunted by identical elements but with opposite sign it can be seen that the transfer matrix between the r *th* and r+*l th* nodes in the network will be follows:

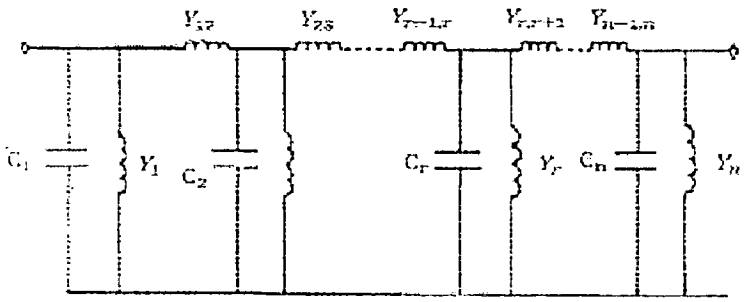
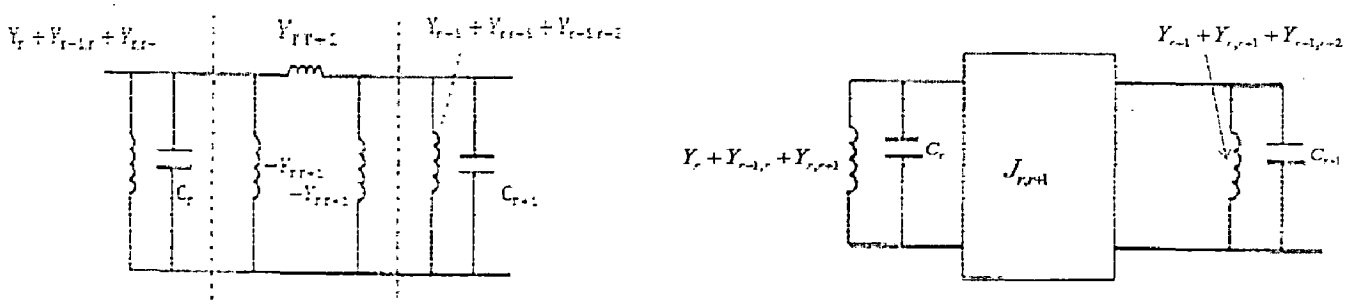


Fig 2.28: Equivalent circuit of the combline filter [13]



(a) (b)

Fig 2.29: Formation of impedance inverters in the combline filters [13]

$$(T)\begin{bmatrix} 1 & 0 \\ j\frac{Y_{r,r+1}}{\tan\theta} & 1 \end{bmatrix}\begin{bmatrix} 1 & j\frac{\tan\theta}{Y_{r,r+1}} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ j\frac{Y_{r,r+1}}{\tan\theta} & 1 \end{bmatrix} = \begin{bmatrix} 0 & j\frac{\tan\theta}{Y_{r,r+1}} \\ j\frac{\tan\theta}{Y_{r,r+1}} & 0 \end{bmatrix} \quad (2.33)$$

Which is the transfer matrix of an admittance inverter of admittance

$$J_{r,r+1} = \frac{Y_{r,r+1}}{\tan\theta} \quad (2.34)$$

2.8. Conclusion

In this chapter we have studied the resonant circuits and the types of coupling between the resonant circuits and the concept of Immitance Inverters and the theory of coupled lines and the modeling of an coupled line by an admittance (J) inverters. Finally theory of tunable combline filter is studied.

Chapter -3

Design of Tunable Combline filter with tunable centre frequency and Variable Bandwidth within the pass band tuning range

3.1. Introduction

This chapter deals with the step by step design procedure of tunable combline band pass filter starting from the design specifications. [14]

Design Tool

The simulation tool used for the design of the tunable filter is Agilent's Advanced Design System (ADS).

Design Procedure

The design method of the tunable filters can be split into three.

1.Design specifications

2.Design parameters

3.Filter Physical dimensions

Design Specifications

Design Specifications required for the filter design are

Frequency band of operation	: 0.5 GHz - 3 GHz
1 dB band width	: 20MHz
Centre frequency tunability step	: 20MHz
Pass band insertion loss	< 2dB
Pass band VSWR	1.5:1
Rejection @ ±80MHz away from centre frequency	: 60dB minimum
Impedance	: 50 ohm

3.2. Converting Band pass filter to normalized Low pass prototype [15, 19]

3.2.1. Calculation of pass band ripple from the specifications:

$$L_{Ar} = -10\log\left(1 - \left(\frac{VSWR-1}{VSWR+1}\right)^2\right) dB \quad (3.1)$$

from the specifications VSWR given is 1.5 . so the pass band ripple is L_{Ar} =0.1772dB

3.2.2. Calculation of the order of the filter

Usually for the bandpass filter the order of the filter is calculated by converting the bandpass specifications to the lowpass specifications using the frequency conversion

$$\omega^1 = \frac{1}{\Delta w}\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right) \quad (3.2)$$

ω^1 is the lowpass equivalent frequency of the bandpass frequency ω

Δw is the equiripple fractional bandwidth

the value of equiripple fractional bandwidth (Δw) = 0.01552 and ω is the rejection @±80MHz away from centre frequency (f_0) .The value of the f_0=1.224GHz and the value of the ω is 1.224GHz + 80MHz =1.304GHz. By putting the value in the equation (1.3) we get ω^1 as 8.16425 .this is the value of Ωs to be substituted in eqn.1.2 .From the values obtained we get the order of the filter as

$$n \geq \frac{\cosh^{-1}\sqrt{\frac{10^{0.1*60}-1}{10^{0.1*0.1772}-1}}}{\cosh^{-1} 8.16425} \geq 3.29$$

from the symmetry of the chebyshef response the order of the filter is choosen as n = 5

3.2.3. Calculation of equiripple fractional bandwidth

$$\text{equi ripple fractional bandwidth } (\Delta w) = \frac{Bandwidth}{f_0} \quad (3.3)$$

where Bandwidth is the equiripple bandwidth

from the given specifications 1dB bandwidth = 20MHz and we have calculated L_{Ar} (pass band ripple) as 0.1772dB,

By using the formula that

$$L_A(\omega^1) = 10\log_{10}(1+\in \cosh^2(n\cosh^{-1}(\frac{\omega^2}{\omega_1^2})) \text{ where } \omega^1 \geq \omega_1^1 \quad (3.4)$$

ω^1= 1dB frequency(which is given in the specifications) and ω_1^1 = edge of the ripple band

$$\in = 10^{\frac{L_{Ar}}{10}} - 1 \quad (3.5)$$

$$= 10^{\frac{0.1778}{10}} - 1 = 0.04164$$

by using the formula (1.5) we get $\frac{\omega^2}{\omega_1^2}$ =1.049

from the calculations equiripple bandwidth is 19.0599 MHz

equi ripple fractional bandwidth (Δw) = 0.01557 since f_0=1.224GHz

3.2.4. Determination of normalised Lowpass prototype filter coefficients

The element values of the two port network of may be computed using the following formulas:

from the specifications of the filter we get the filter coefficients as

g_0=1

g_1= 1.30178

g_2= 1.3455

g_3= 2.12879

g_4=1.3455

g_5=1.30178

g_6= 1

3.2.5. Determination of Bandpass Design parameters from the normalized lowpass prototype [16-17]

Calculation of Coupling Coefficients

$$M_{1,2}=\frac{\Delta w}{\sqrt{g_1 g_2}} = 0.01172 \qquad \text{where } \Delta w \text{ is the fractional bandwidth} \tag{3.6}$$

$$M_{2,3}=\frac{\Delta w}{\sqrt{g_2 g_3}} = 0.00917 \tag{3.7}$$

$$M_{3,4}=\frac{\Delta w}{\sqrt{g_3 g_4}} = 0.00917 \tag{3.8}$$

$$M_{4,5}=\frac{\Delta w}{\sqrt{g_4 g_5}} = 0.01172 \tag{3.9}$$

Calculation of external quality factor

$$Q_{e1} = Q_{es} - \frac{g_0 g_1}{\Delta w} = 83.8 \tag{3.10}$$

3.2.6. Implementation of Coupling Coefficients for determining the spacing between the resonators using EM simulator [18, 25]

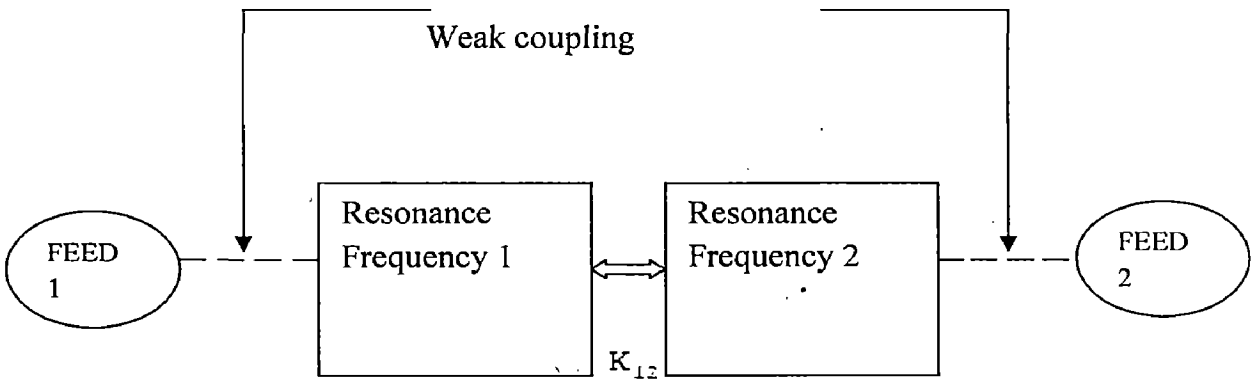


Fig 3.1 :Extraction of coupling coefficients between resonators using EM simulation

If we loosely couple to the pair of resonators tuned to same frequency we get the double peak response shown in figure

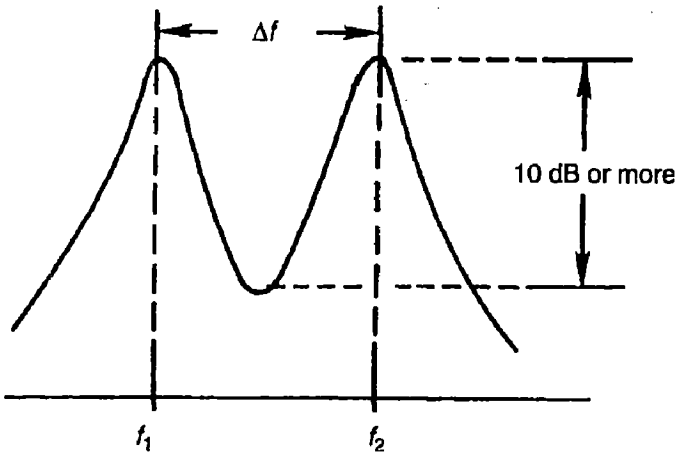


Fig. 3.2: Frequency response of double tuned resonator pair [20]

The null between two peaks should be around 20 dB more to guarantee loose coupling between the resonators.

The expression for coupling coefficients is [20]

$$K = \frac{f_{HIGH}-f_{LOW}}{f_0} = \frac{f_{HIGH}^2-f_{LOW}^2}{f_{HIGH}^2+f_{LOW}^2} \qquad (3.11)$$

f_{HIGH} is the frequency of Higher Peak and f_{LOW} is the frequency of Lower peak

The values of spacing for the required coupling coefficients is shown in figure

spacing between resonator 1 and resonator 2 is 6.9 mm which is equal to the spacing between the resonator 4 and resonator 5 ,spacing between resonator 2 and resonator 3 is 7.6mm which is equal to the spacing between the resonator 3 and resonator 4.

3.2.7. Implementation of External quality factor for determining the taplength position of the input and output feed at the first and last resonators using EM simulator [21]

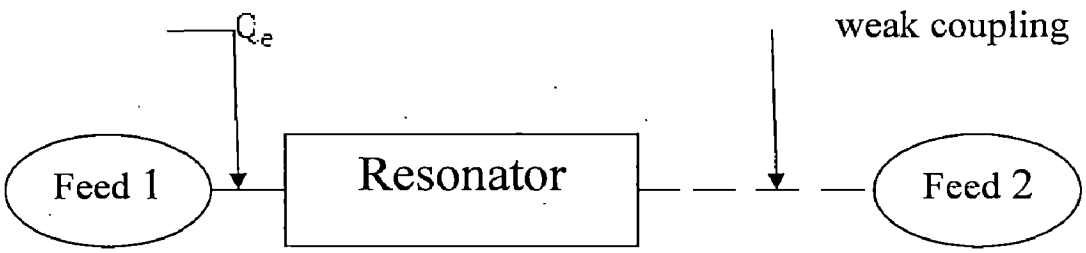


Fig. 3.3: Extracting (Q_e) using EM simulator

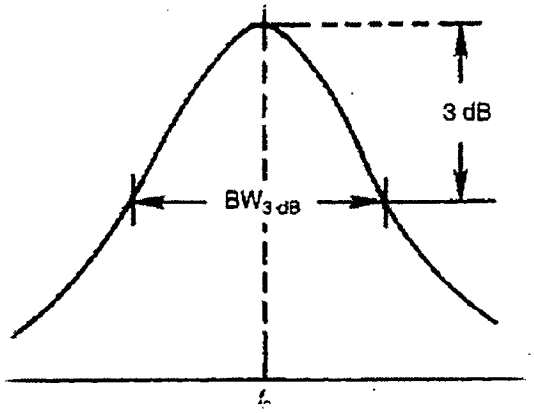


Fig 3.4 : Frequency response of singly tuned resonator

Q_e can be calculated by using the formula : $Q_e = \frac{f_0}{3\ dB\ bandwidth}$ (3.12)

3.3. Design of Tunable Combline Band pass filter [22-23]

The general specifications required for the tunable filter are

1. The **TUNING RANGE** or **TUNING BANDWIDTH** ($[f_1\ f_2]$): the range of frequencies in which the filter should be tuned.

2. **INSTANTANEOUS BANDWIDTH range** ($[\Delta B_{Lmin},\ \Delta B_{Lmax}]$) the filter has within the tuning range, given for a attenuation level (L dB).

3. Maximum **VSWR in** the filter pass band.

4. **CAPACITANCE RANGE** ($[C_{smin}, C_{smax}]$) of the tunable elements used.

3.3.1. Determination of the filter electrical design parameters

Electrical design Parameters of the filter are:

1. Design center frequency (f_0)
2. Resonator electrical length (θ_0) at f_0
3. Instantaneous bandwidth (ΔB_{L0})

4. Tuning device capacitance (C_{s0}) at f_0

Center frequency (f_0)

Design center frequency (f_0): $f_0 = \sqrt{f_1 . f_2}$ (3.13)

From the given specifications $f_0 = \sqrt{0.5 * 3} = 1.224$ GHz

Resonator electrical length (θ_0) at f_0

The instantaneous bandwidth dependence on the resonator electrical length (θ) is given by

$$\Delta B_L = k \frac{\theta . \tan(\theta)}{\tan(\theta) + \theta . (1 + \tan(\theta)^2)} \quad (3.14)$$

K is constant depends on the resonator geometry

[13] Defining $\Gamma = 10 \log (\Delta B_{Ln}) = 10 \log \left(\frac{\theta . \tan(\theta)}{\tan(\theta) + \theta . (1 + \tan(\theta)^2)} \right)$ (3.15)

Γ: normalized instantaneous bandwidth

Plotting Γ and θ in matlab we get the graph as. θ_{max} occurs at around 53 degrees

Since the typical specification consists of maintaining the instantaneous bandwidth within a range, ([ΔB_{Lmin} , ΔB_{Lmax}]) .Then Γ is useful to select (θ_0)

Defining $\Delta \Gamma = 10 \log \frac{\Delta B_{Lmax}}{\Delta B_{Lmin}}$ (3.16)

where ([ΔB_{Lmin} , ΔB_{Lmax}]) = ([20MHz 50MHz])

where $\Delta \Gamma = 3.979$

$\Gamma (\theta_{min1} , \theta_{min2}) = \Gamma (\theta_{max}) - \Delta \Gamma$

Then the range of values of θ_0 is $\theta_{min1} \frac{f_0}{f_1} \leq \theta_0 \leq \theta_{min2} \frac{f_0}{f_2}$ (3.17)

θ_{min1} =14.4 degrees and θ_{min2} = 82.6 degrees which are obtained from MATLAB plot.

Then the range of values of θ_0 is $35.79 \leq \theta_0 \leq 37.44$

θ_c is selected as 36 degrees

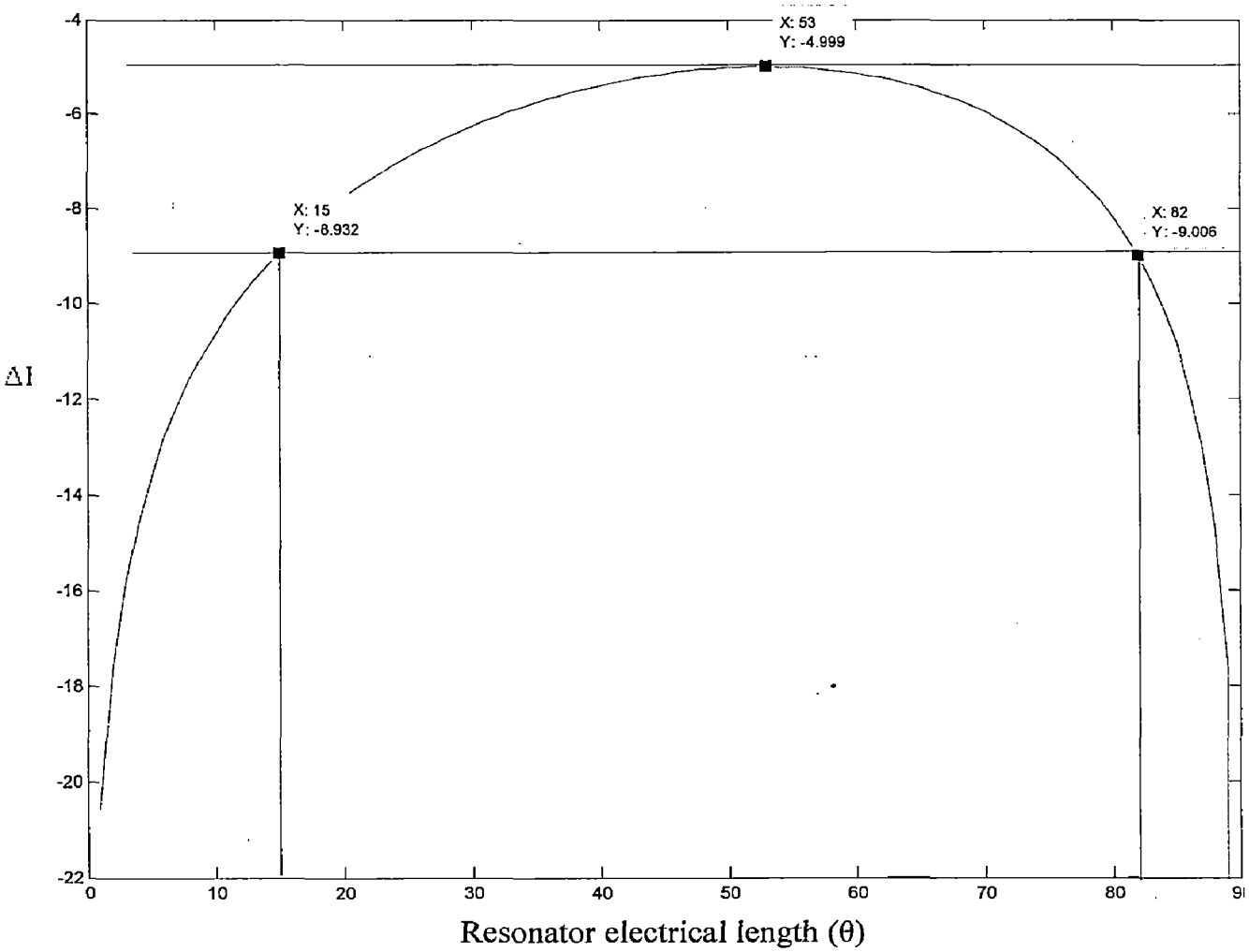


Fig: 3.5 Plot of Normalized Instantaneous Bandwidth Vs resonator electrical length

Instantaneous Bandwidth (ΔB_{L0})

$$\frac{\Delta B_{Lmin}}{\min_{\theta\in[\theta_1,\theta_2]}\{\Delta B_{Ln}(\theta)\}}\Delta B_{Ln}(\theta_0) \leq \Delta B_{L0} \leq \frac{\Delta B_{Lmax}}{\Delta B_{Ln}(\theta_{max})}\Delta B_{Ln}(\theta_0) \qquad (3.18)$$

As mentioned previously ΔB_{Lmin} = 20MHz and ΔB_{Lmax}=50MHz and θ_{max} occurs at 53 degrees.from the calculations we have choosen θ_0 = 36 degrees and $\Delta B_{Ln}(\theta_0)$ is calculated as 0.2760 and the value of $\Delta B_{Ln}(\theta_{max})$ is 0.31626 and θ_1, θ_2 satisfy the relation

$$\theta_{min1} \leq \theta_1 \leq \theta_0 \leq \theta_2 \leq \theta_{min2} \qquad (3.19)$$

Note: In substituting the values of the θ_0 in the equations, θ_0 should be in radians.

Range of values of ΔE_{L0} is $20.8MHz \leq \Delta B_{L0} \leq 43.63$ MHz

ΔB_{L0} is choosen as 30MHz

Where $\Delta B_{Ln}(\theta) = \frac{\theta.\tan(\theta)}{\tan(\theta)+\theta.(1+\tan(\theta)^2)}$

Tuning Device Capacitance (C_{s0}) at f_0

The equation that relates a tuning frequency and its corresponding tuning capacitance is

$$\frac{1}{Y_s}.2\pi f.\tan\theta.C_s(f)=1 \tag{3.20}$$

θ is the resonator electrical length at the tuning frequency f , Y_s is the resonator line admittance which is choosen by the designer to fix the admittance level within the filter. The tuning device provides a variable capacitance within a range ($[C_{smin}, C_{smax}]$). The smallest capacitance tunes the highest tunable frequency, f_{Mtun} and the largest capacitance tunes the lowest one, f_{mtun}. $C_{s0}(f)$ is the capacitance at the centre frequency.

The maximum tunable frequency can be expressed as an implicit function of C_{s0} by means of

$$f_{Mtun}\, C_{smin}\, \tan\left(\frac{\theta_0}{f_0} f_{Mtun}\right) = f_0 C_{s0} \tan\theta_0 \tag{3.21}$$

The minimum tunable frequency can be expressed as an implicit function of C_{s0} by means of

$$f_{mtun}\, C_{smax}\, \tan\left(\frac{\theta_0}{f_0} f_{mtun}\right) = f_0 C_{s0} \tan\theta_0 \tag{3.22}$$

the range of ($[C_{smin}, C_{smax}]$) taken is [0.2pF 2.7pF].

for the value of C_{s0} we plot the above two equations in MATLAB shown

the range of possible values of C_{s0} is given by

$$\frac{f_2.\tan\left(\frac{\theta_0.f_2}{f_0}\right)}{f_0.\tan\theta_0} C_{smin} \leq C_{s0} \leq \frac{f_1.\tan\left(\frac{\theta_0.f_1}{f_0}\right)}{f_0.\tan\theta_0} C_{smax} \tag{3.23}$$

f_2 = 3GHz , f_1=0.5GHz , f_0=1.224GHz, θ_0= 36 degrees , ($[C_{smin}, C_{smax}]$) = [0.2pf 2.7pf]

The range of possible values are $0.172 \leq C_{sj} \leq 0.878$ .The value of capacitance choosen is 0.75pF.

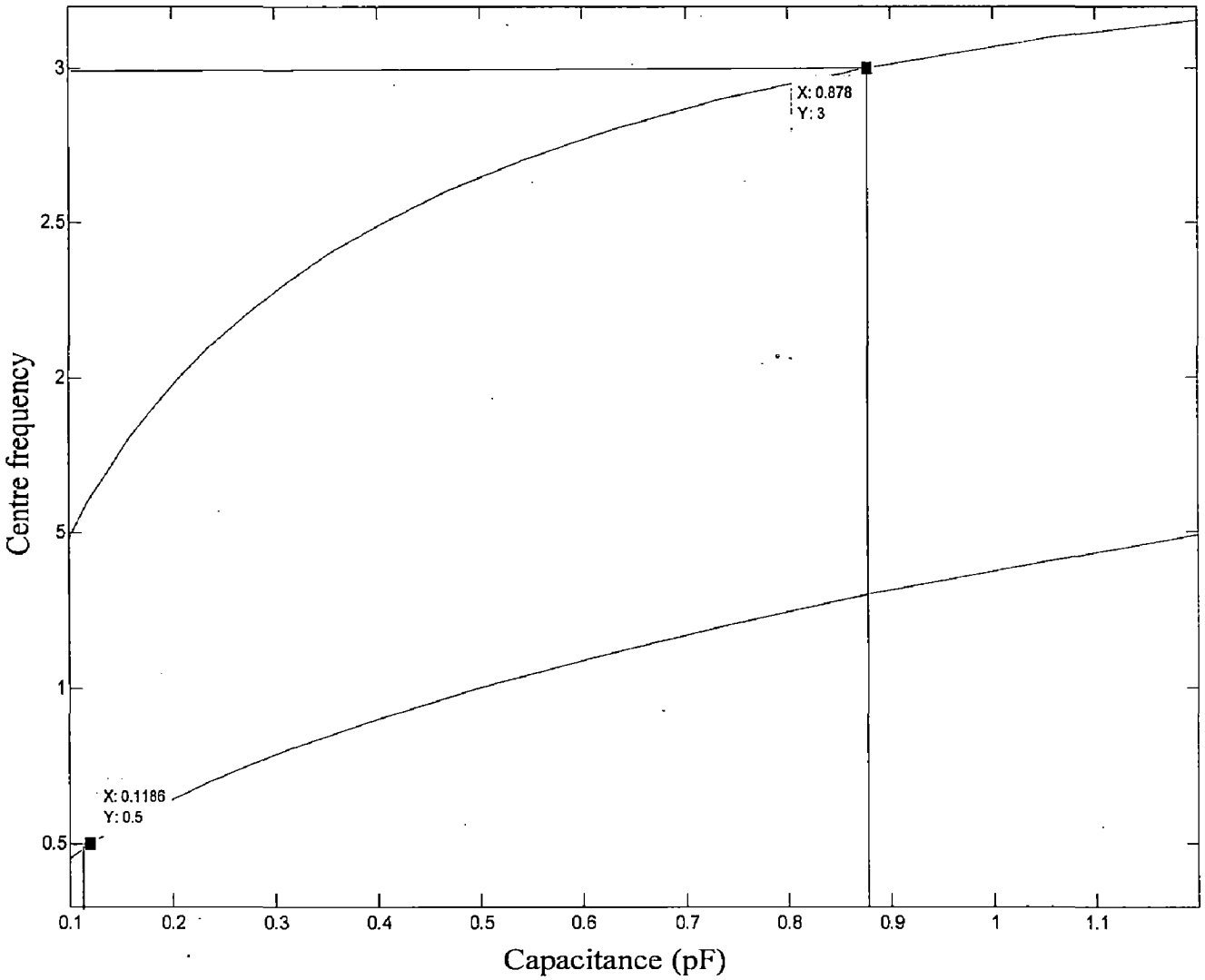


Fig 3.6 Plot of tuning device capacitance Vs tuning frequency

3.4. Filter Physical Dimensions

HYBRID -MIC tunable combline filter design using ADS(Advanced Design System)

The substrate used for the design of tunable combline filter is

$\varepsilon_r = 3.38$

Thickness of dielectric =1.524mm

copper thickness =9μm

Dissipation factor =0.0024

3.4.1. Calculation of width and length of the resonators

By using Line Calc in ADS (Advanced Design Systems) the width and length of the microstrip lines are

width = 4.2 mm for 44.565 ohm filter internal impedance

length =12.4296mm for an electrical length of 36 degrees

width =3.512mm for 50 ohm feed line.

The resonator line admittance , is choosen to fix the admittance level within the filter.

3.4.2. Calculation of Spacing between the resonators by using parametric extraction of coupling coefficients using EM simulator

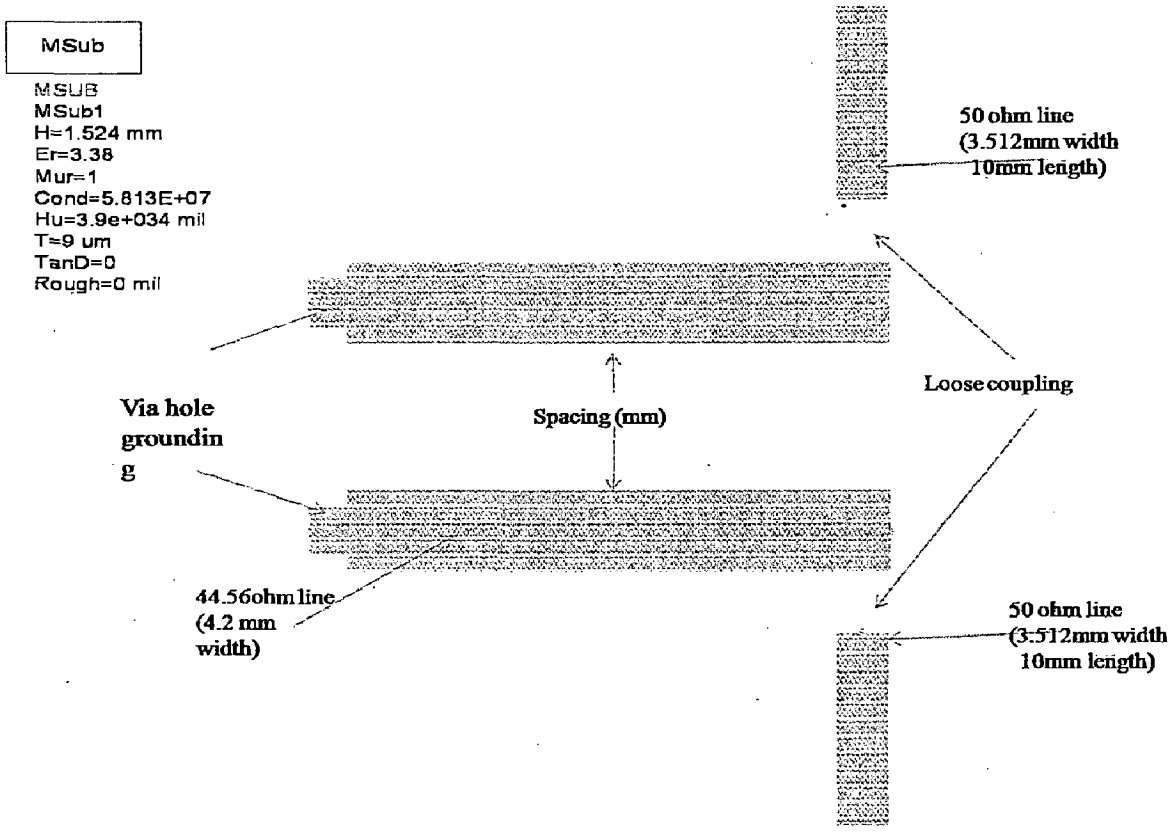


Fig 3.7 : structure for extracting the coupling coefficients between resonators in ADS

spacing between resonator 1 and resonator 2 is 6.9 mm which is equal to the spacing between the resonator 4 and resonator 5 ,spacing between resonator 2 and resonator 3 is 7.6mm which is equal to the spacing between the resonator 3 and resonator 4.

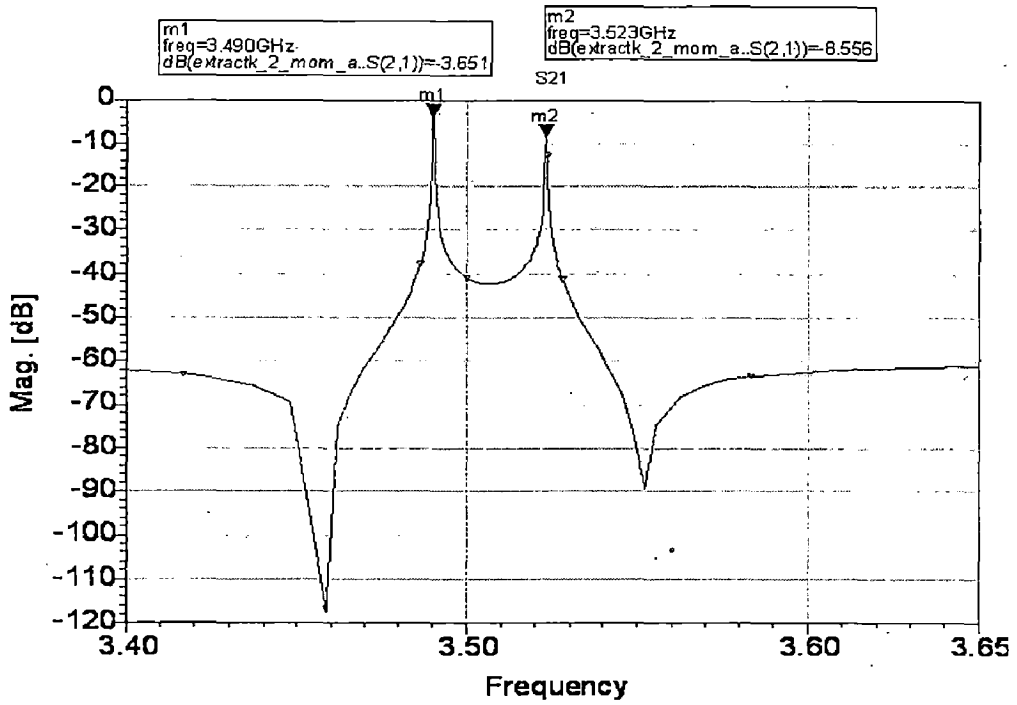


Fig 3.8 Spacing between the resonators is found by extracting the K value

K value is given by $K = \frac{f_{HIGH}-f_{LOW}}{f_0} = \frac{f_{HIGH}^2-f_{LOW}^2}{f_{HIGH}^2+f_{LOW}^2}$ (3.24)

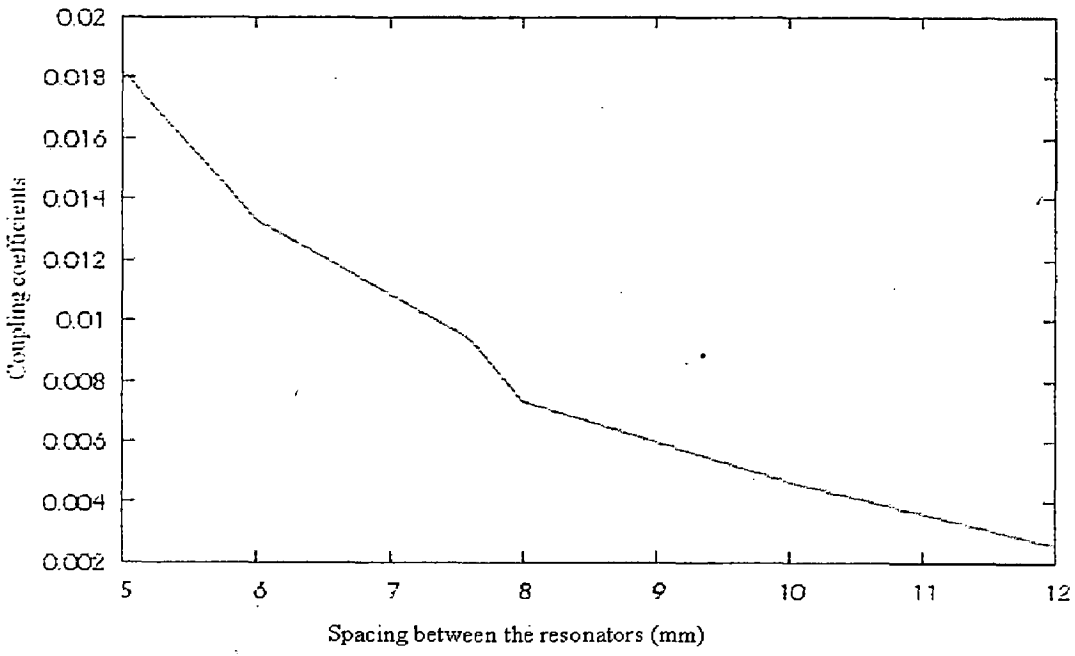


Fig 3.9: Graph extacted between Spacing of the resonators and coupling coefficients

3.4.3. Calculation of taplength position by using parametric extraction of external quality factor using EM simulator

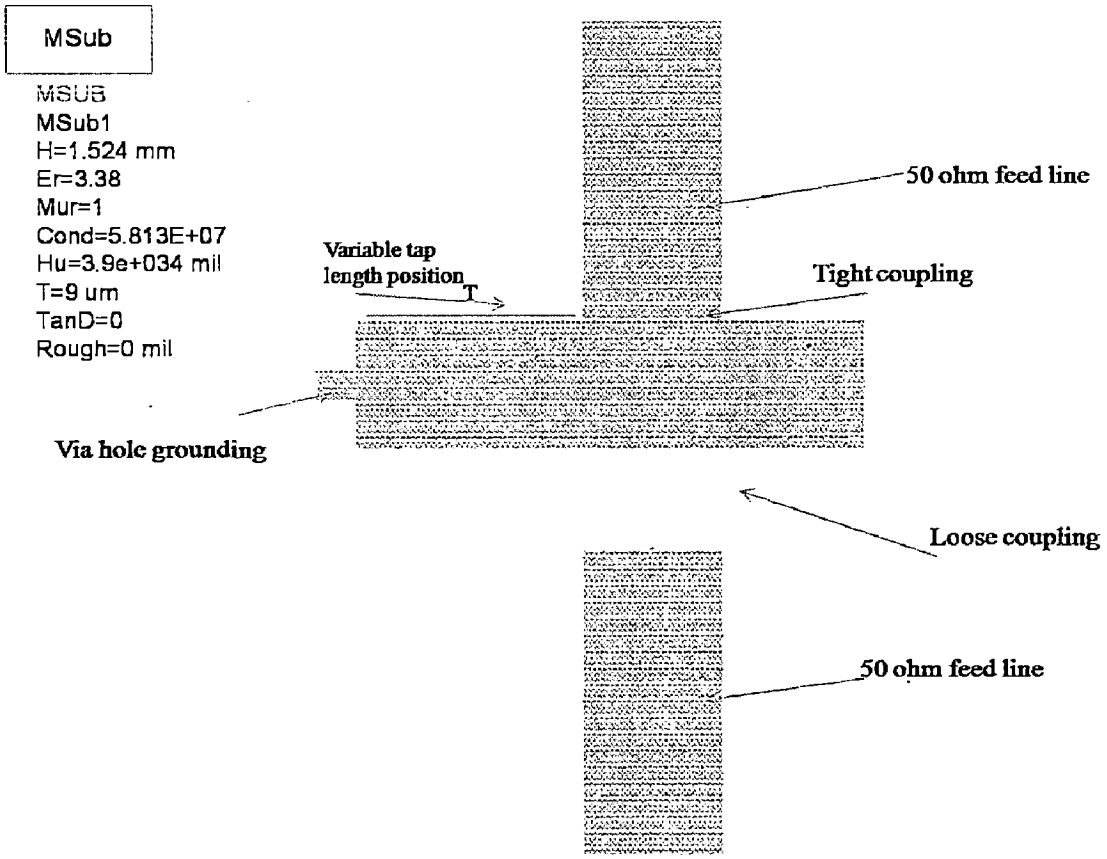


Fig 3.10 Structure for extracting external quality factor using ADS

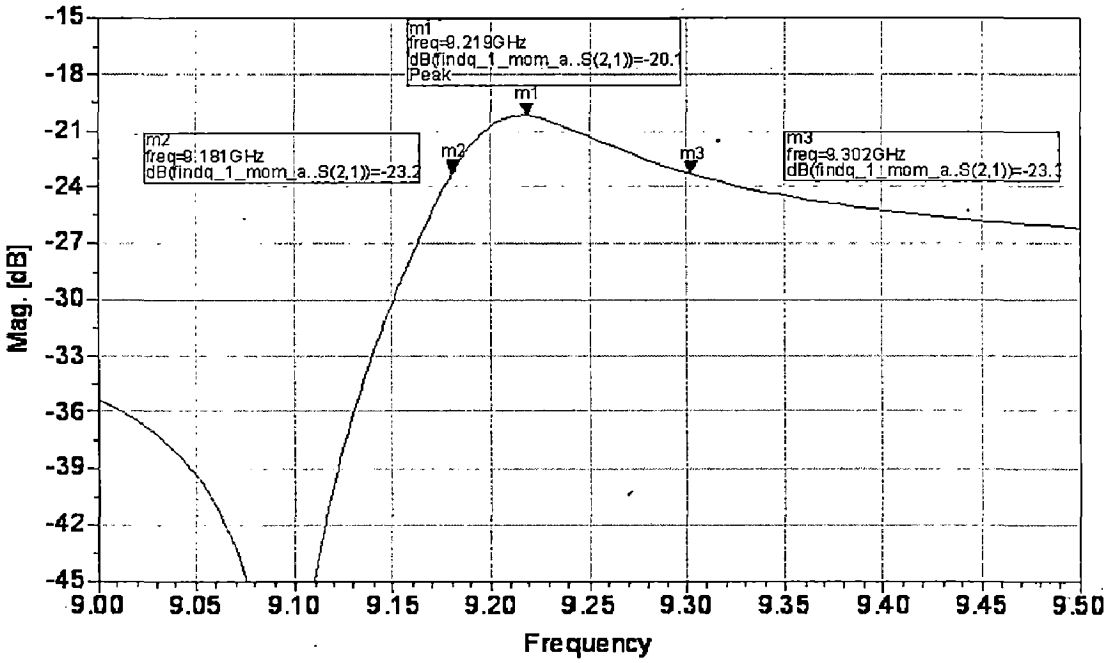


Fig 3.11 : Taplength position for the required value of external quality factor is 7.5mm

3.5. Layout of the filter and simulation results

Design Tool : The simulation tool used for the design of the tunable filter is Agilent's Advanced Design System (2.5D EM simulator)

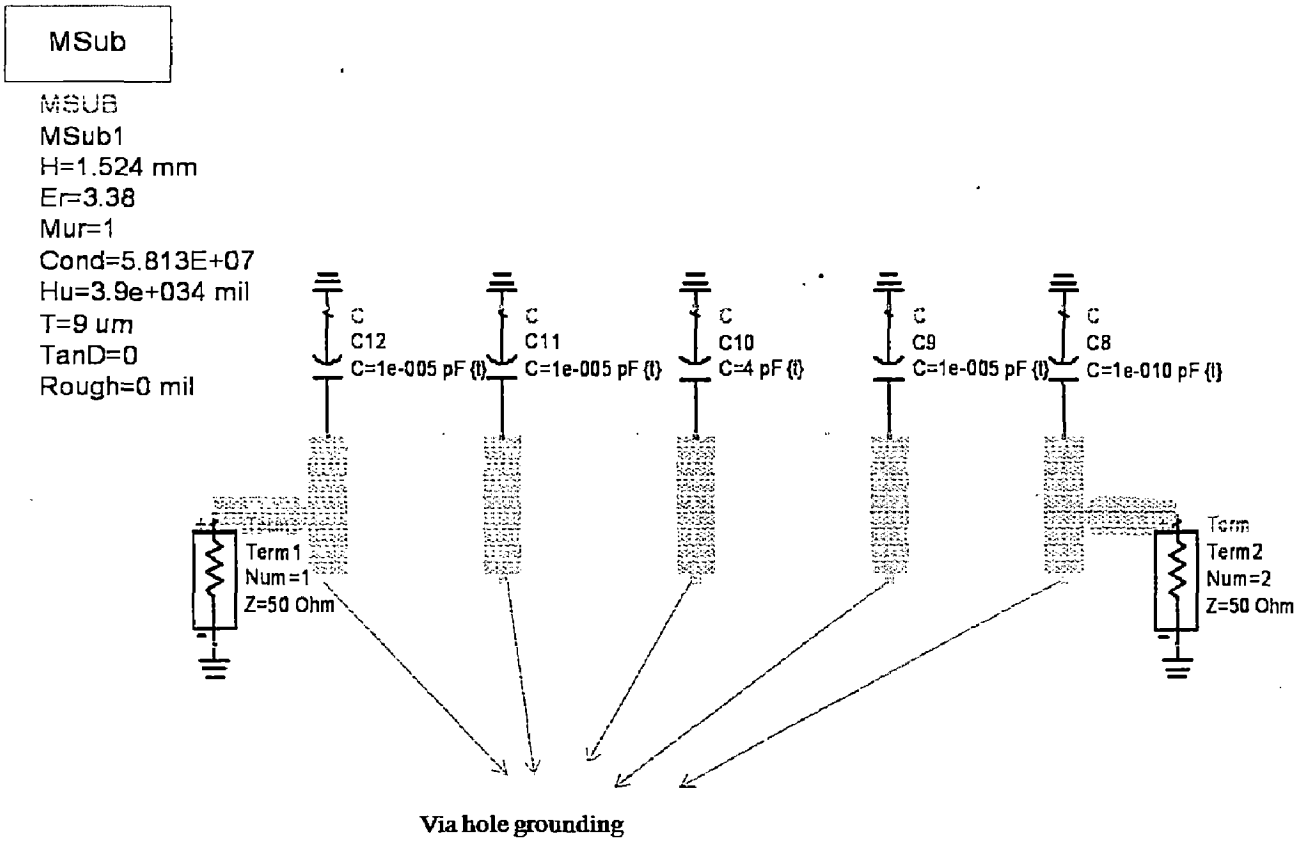


Fig 3.12: Layout of the filter in ADS including Via Hole Groundings [24]

Capacitance Values Required For Tuning The Filter:

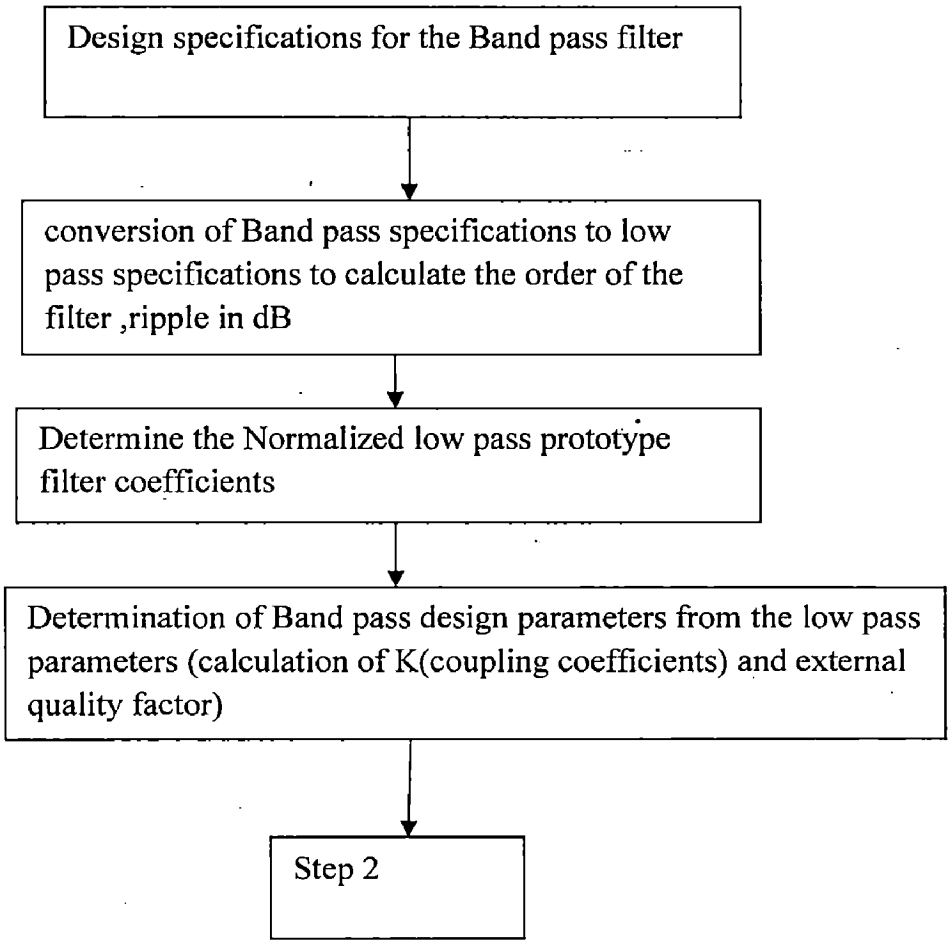
Capacitance Values (C1,C2,C3,C4,C5) (pF)	Centre Frequency (GHz)	Insertion Loss (dB)	Rejection@60dB from Centre Frequency (MHz)	Return Loss (dB)	1-dB Bandwidth (MHz)
21.2,22,2,22,22	0.553	-0.111	46.2	-15.944	4
18.2,19,2,19,19	0.595	-0.046	38	-19.946	2
13.2,14,2,14,14	0.689	-0.111	24	-15.002	2
10.2,11,2,11,11	0.773	-0.150	24	-14.019	2.8
8.2,9,0.1,9,9	0.850	-0.1	22	-15.221	4
6.3,7,0.1,7,7	0.955	-0.061	30	-17.161	5

Capacitance Values (C1,C2,C3,C4,C5) (pF)	Centre Frequency (GHz)	Insertion Loss (dB)	Rejection@60dB from Centre Frequency (MHz)	Return Loss (dB)	1-dB Bandwidth (MHz)
5.2,6,0.1,6,6	1.026	-0.001	34	-28.4	6
4.6,5,0.1,5,5	1.114	-0.085	40	-16.617	9
4.1,4.5,0.1,4.5,4.5	1.167	-0.008	38	-25.838	5
3.6,4,0.1,4,4	1.229	-0.032	35	-23.227	8
3.2,3.5,0.1,3.5,3.5	1.302	-0.027	53	-20.408	6
3,3.3,0.1,3.3,3.3	1.335	-0.01	42	-16.297	5
2.85,3.1,0.1,3.1,3.1	1.373	-0.04	40	-16.108	5
2.7,2.7,0.1,2.7,2.7	1.454	-0.003	33	-18.109	3
2.3,2.3,0.1,2.3,2.3	1.548	-0.005	88	-29.132	4
2.1,2.1,0.1,2.1,2.1	1.605	-0.006	40	-24.418	4
1.5,1.9,4.5,1.9,1.9	1.667	-0.008	22	-20.636	3
1.2,1.7.4.5,1.7,1.7	1.737	-0.033	24	-17.034	2.8
1.2,1.5,4.5,1.5,1.5	1.816	-0.047	28	-21.008	3
0.9,1.3,4,1.3,1.3	1.906	-0.083	29	-21.743	2.8
0.7,1.1,4,1.1,1.1	2.011	-0.176	32	-14.402	2.8
0.6,0.9,4,0.9,0.9	2.134	-0.149	30	-14.7	3
0.5,0.7,4,0.7,0.7	2.279	-0.048	28	-18.387	3
0.2,0.5,4,0.5,0.5	2.456	-0.049	20	-21.047	2.8
0.1,0.3,4,0.3,0.3	2.669	-0.115	24	-17.642	2.9
0.1,0.1,4,0.1,0.1	2.934	-0.061	36	-17.713	2.8

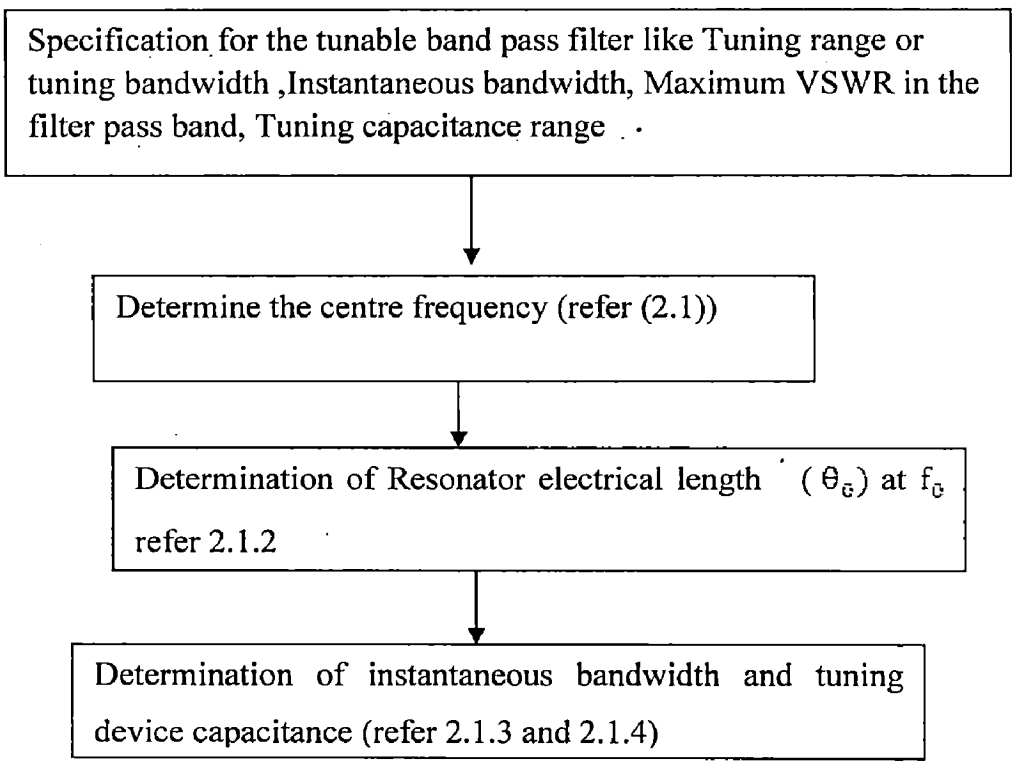
Table 3.1: Capacitance Values Required For Tuning The Filter

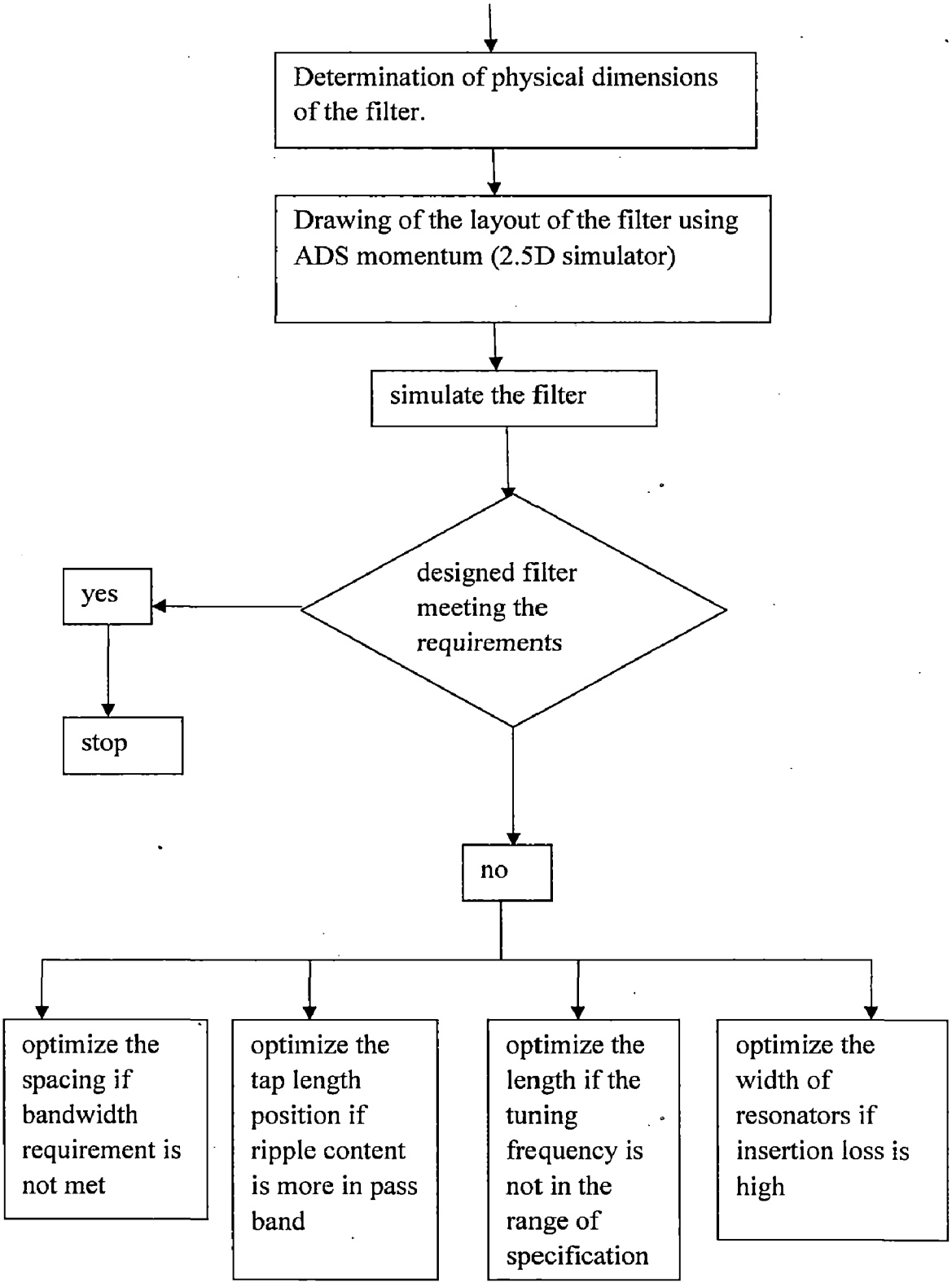
Flowchart for the Tunable Combline Filter Design

STEP 1: Determination of basic prototype filter parameters from the given specifcations

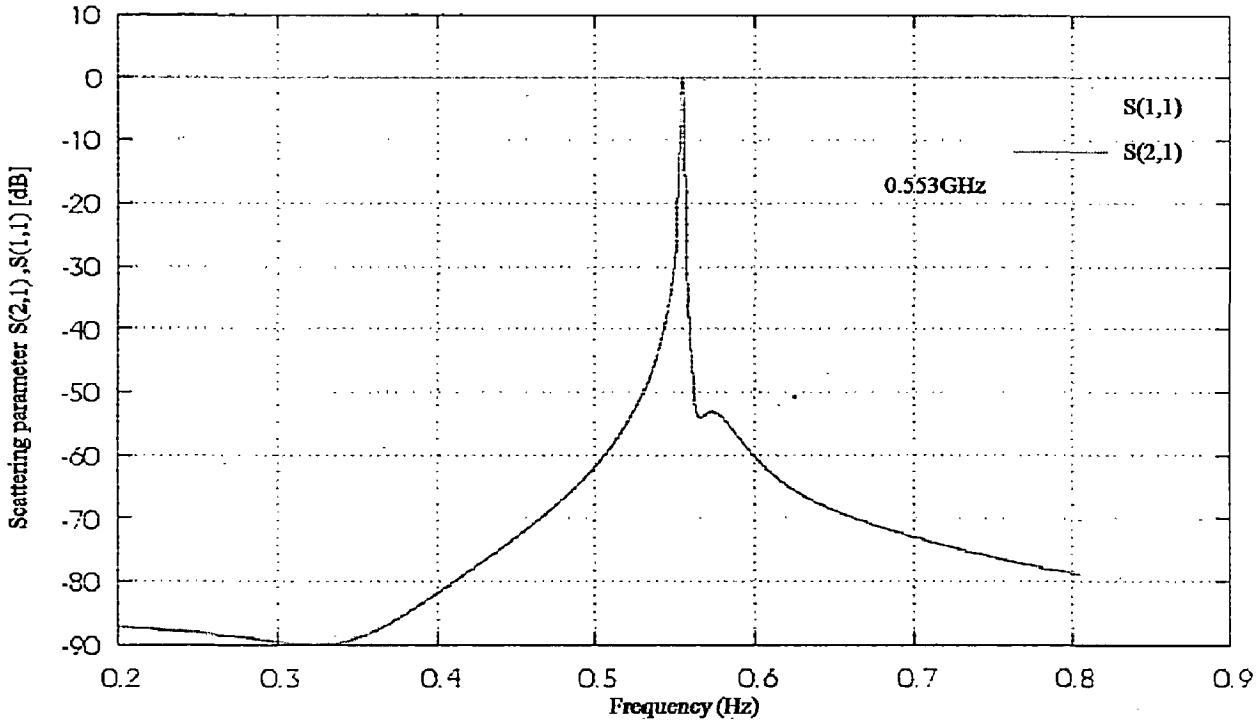


STEP 2: Design of tunable combline band pass filter

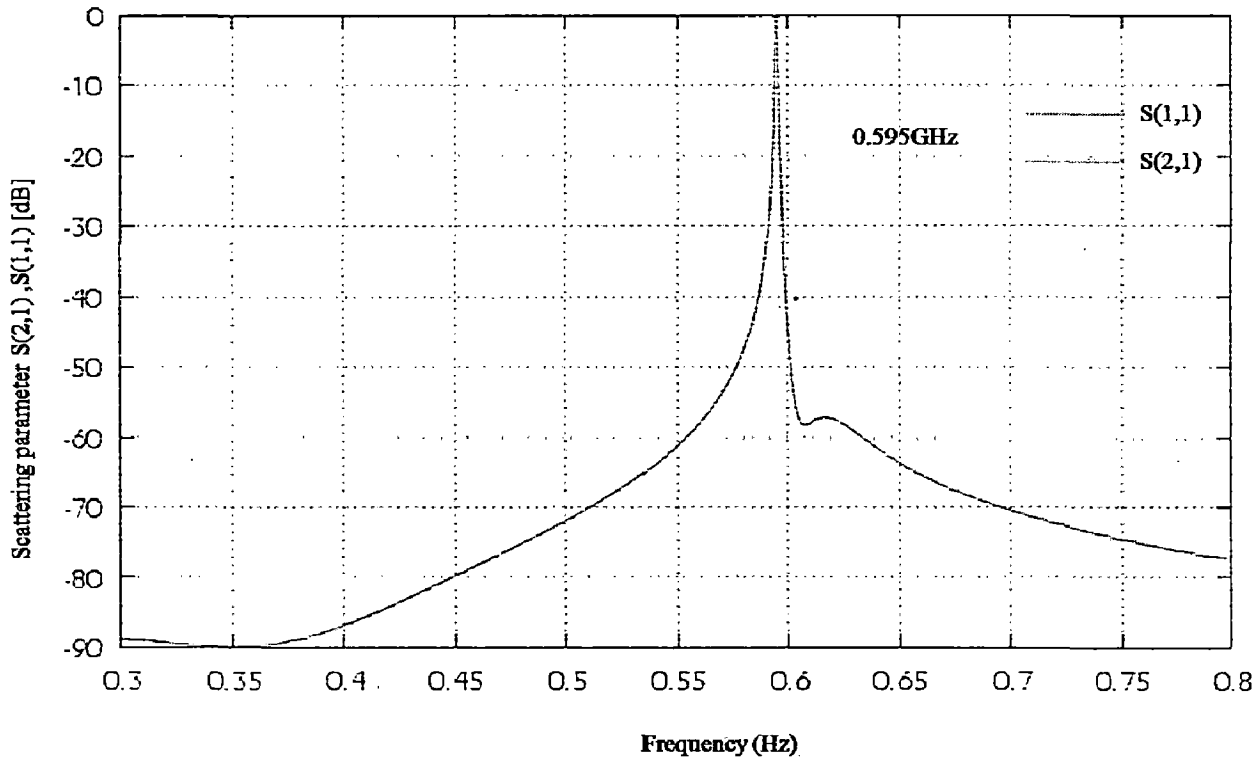



Determination of physical dimensions of the filter.
Drawing of the layout of the filter using ADS momentum (2.5D simulator)
simulate the filter
designed filter meeting the requirements
yes
stop
no
optimize the spacing if bandwidth requirement is not met
optimize the tap length position if ripple content is more in pass band
optimize the length if the tuning frequency is not in the range of specification
optimize the width of resonators if insertion loss is high

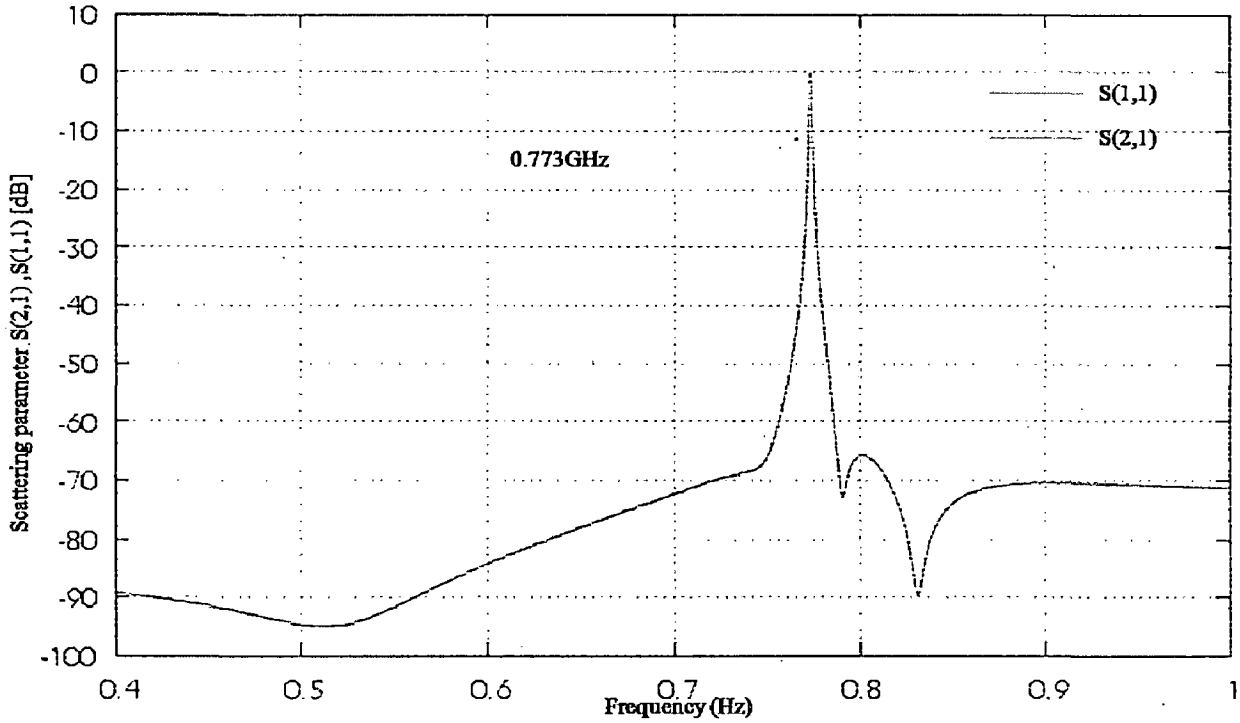
Simulation Results With Ideal Tunable Capacitors:



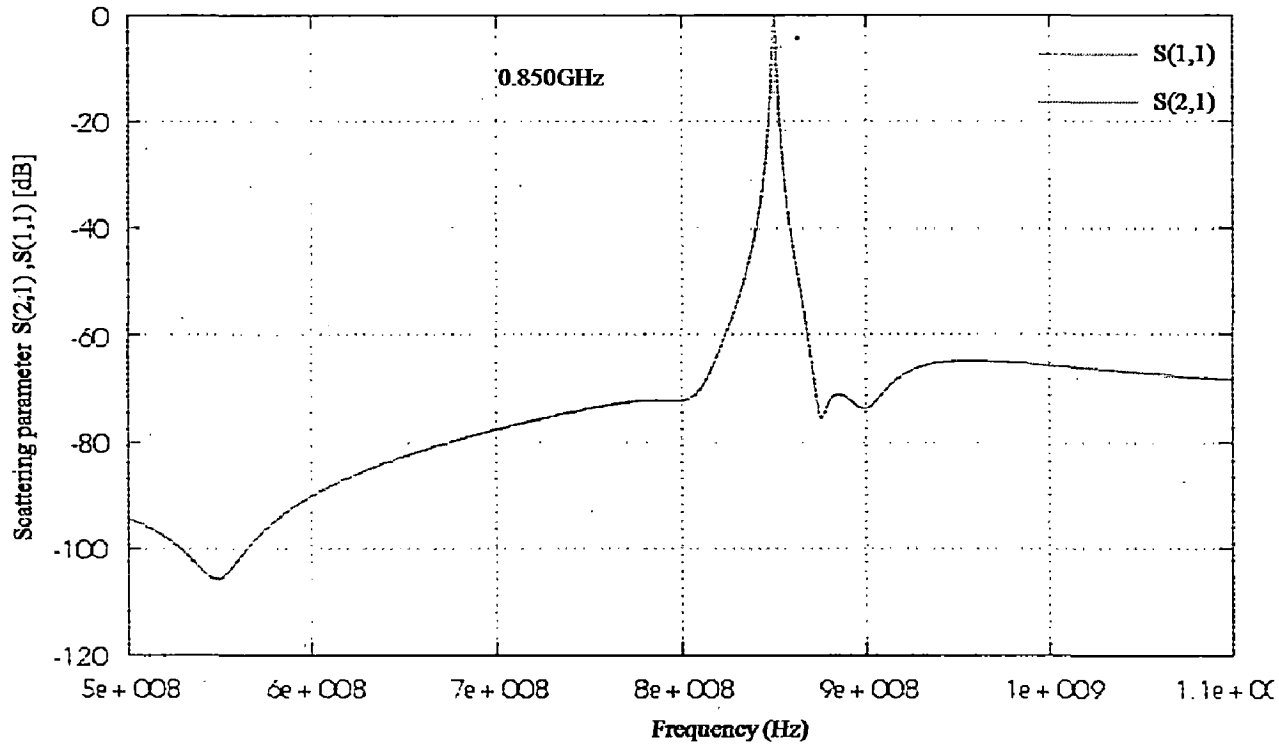
(a)



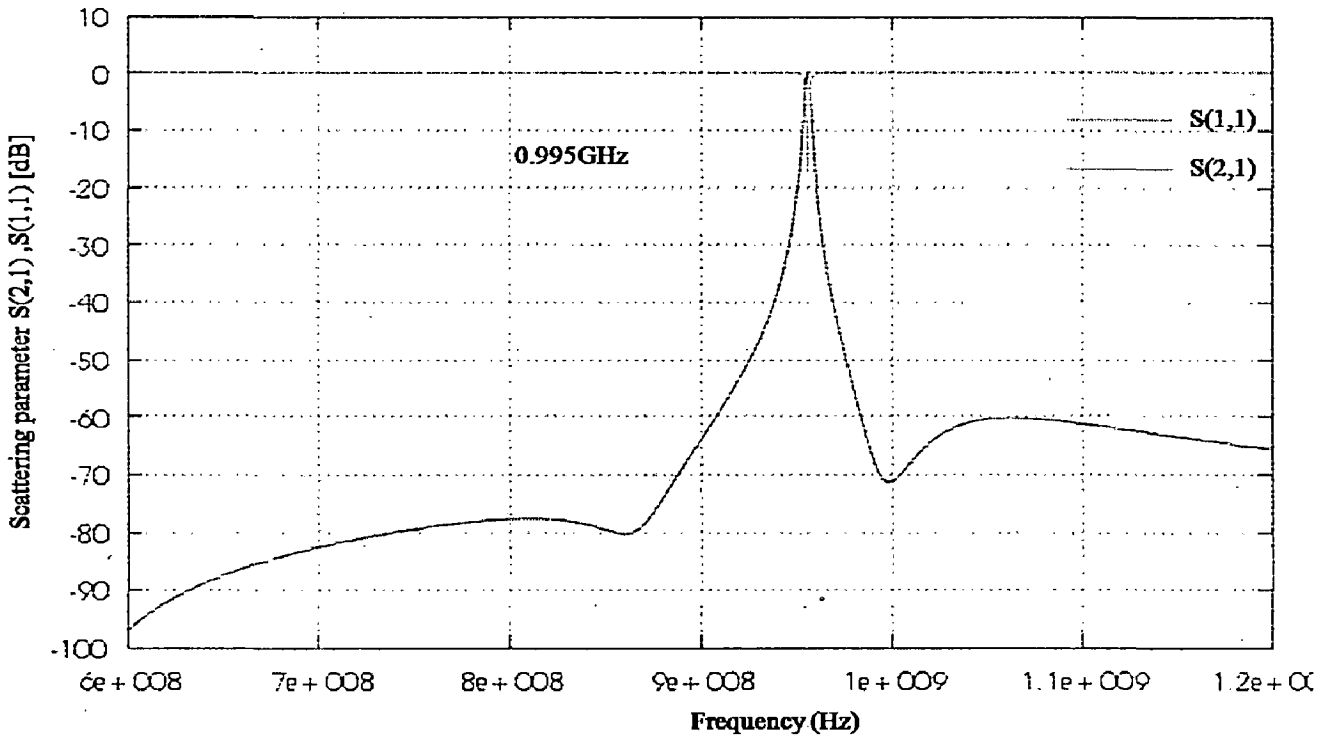
(b)


0.773GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
0.4
0.5
0.6
0.7
0.8
0.9
1

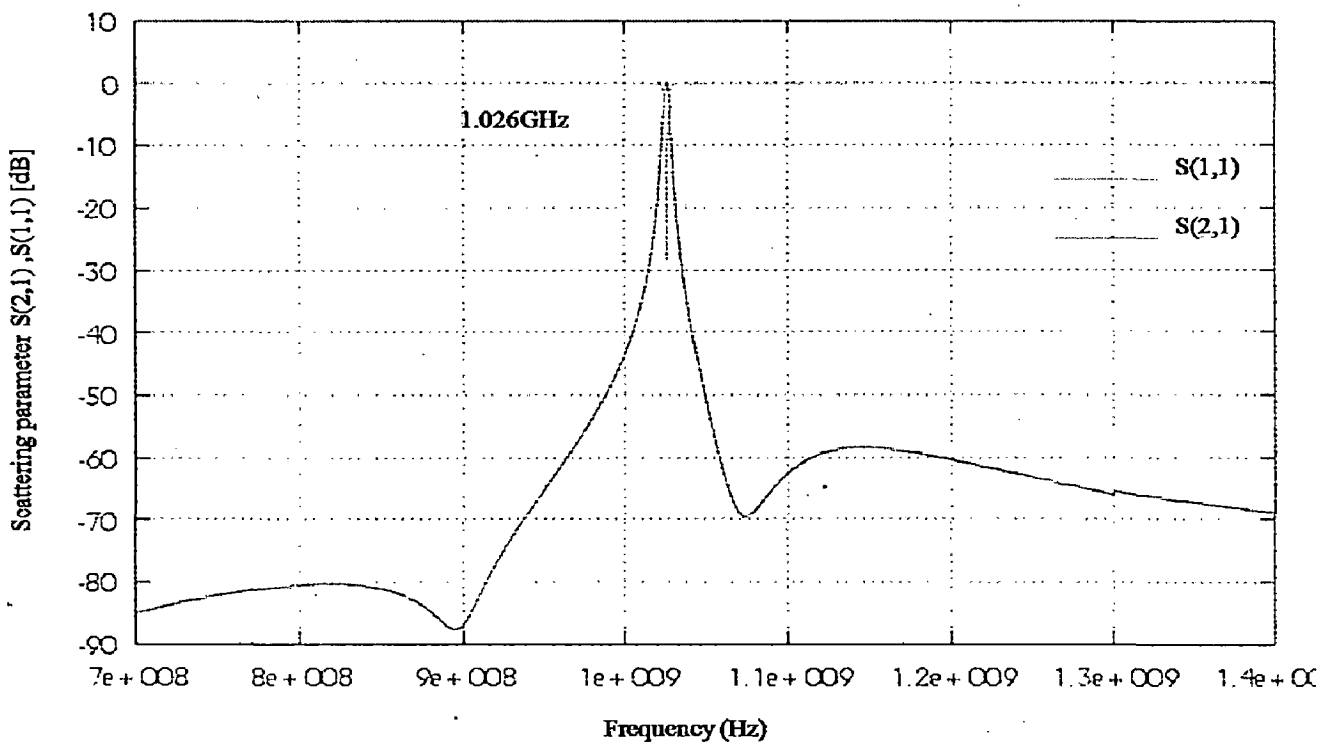
(c)


0.850GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
0
-20
-40
-60
-80
-100
-120
5e+008
6e+008
7e+008
8e+008
9e+008
1e+009

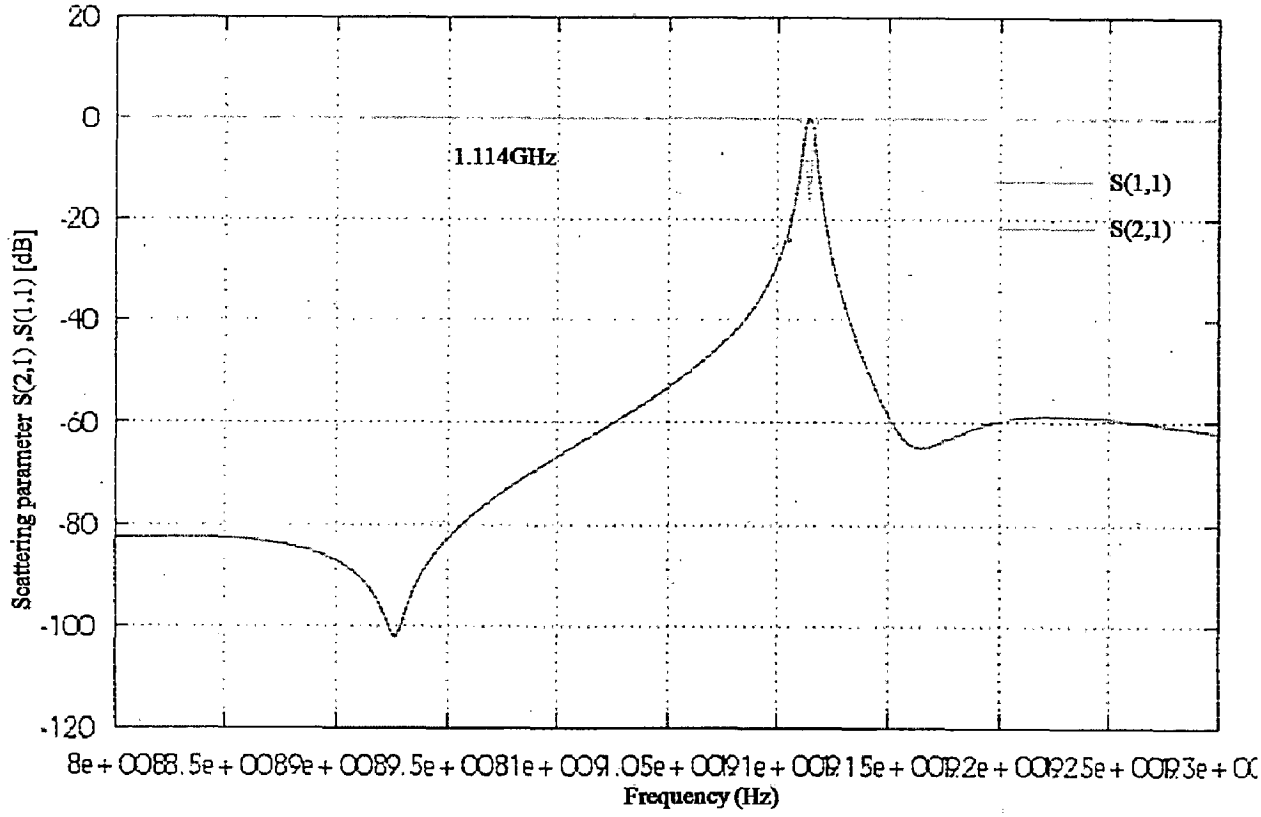
(d)


0.995GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
6e+008
7e+008
8e+008
9e+008
1e+009
1.1e+009
1.2e+0

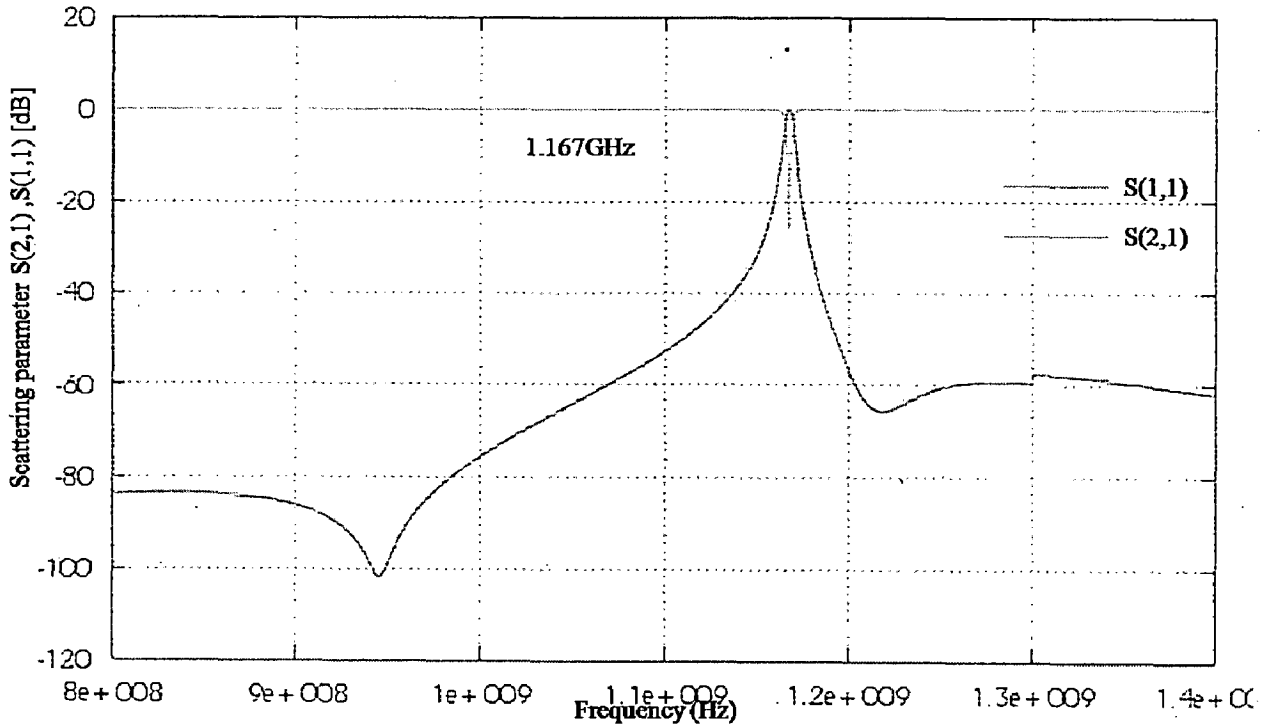
(e)


1.026GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
7e+008
8e+008
9e+008
1e+009
1.1e+009
1.2e+009
1.3e+009
1.4e+0

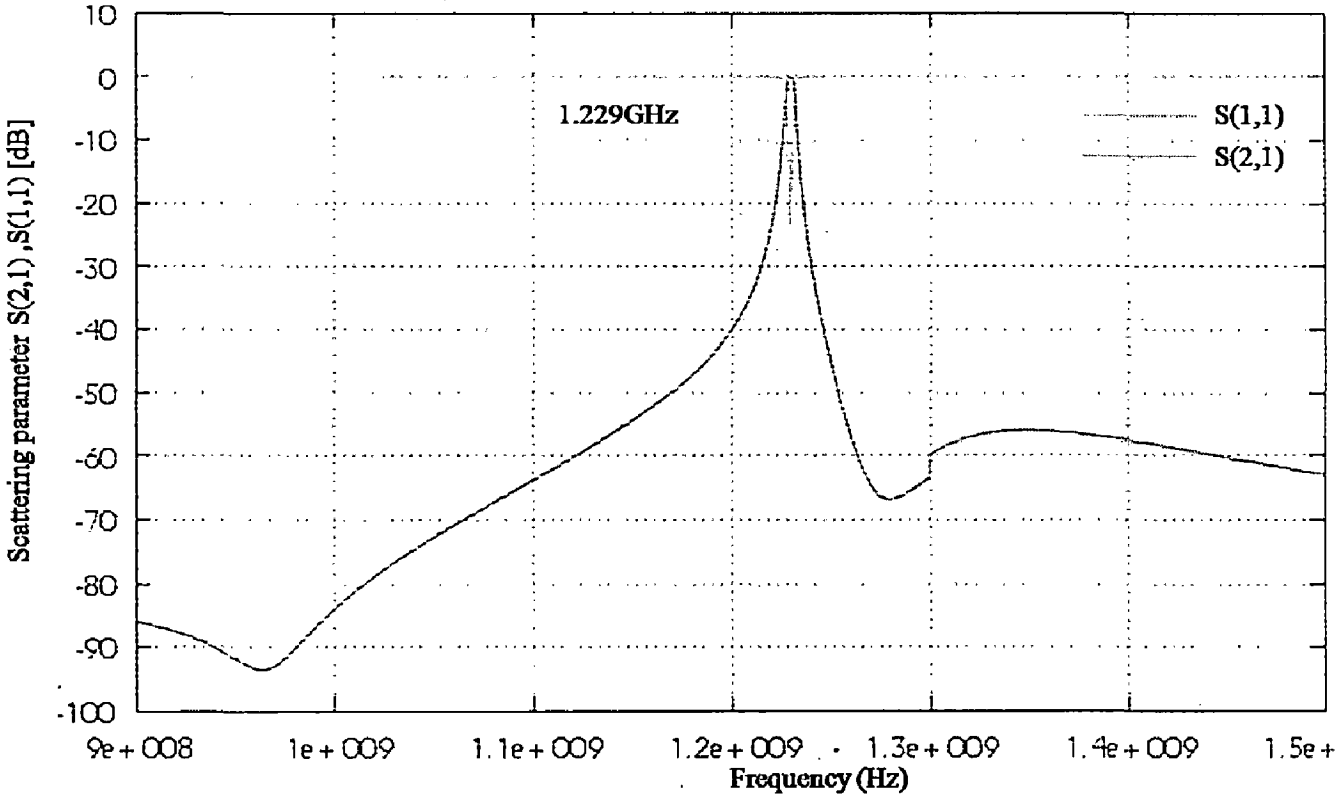
(f)


1.114GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

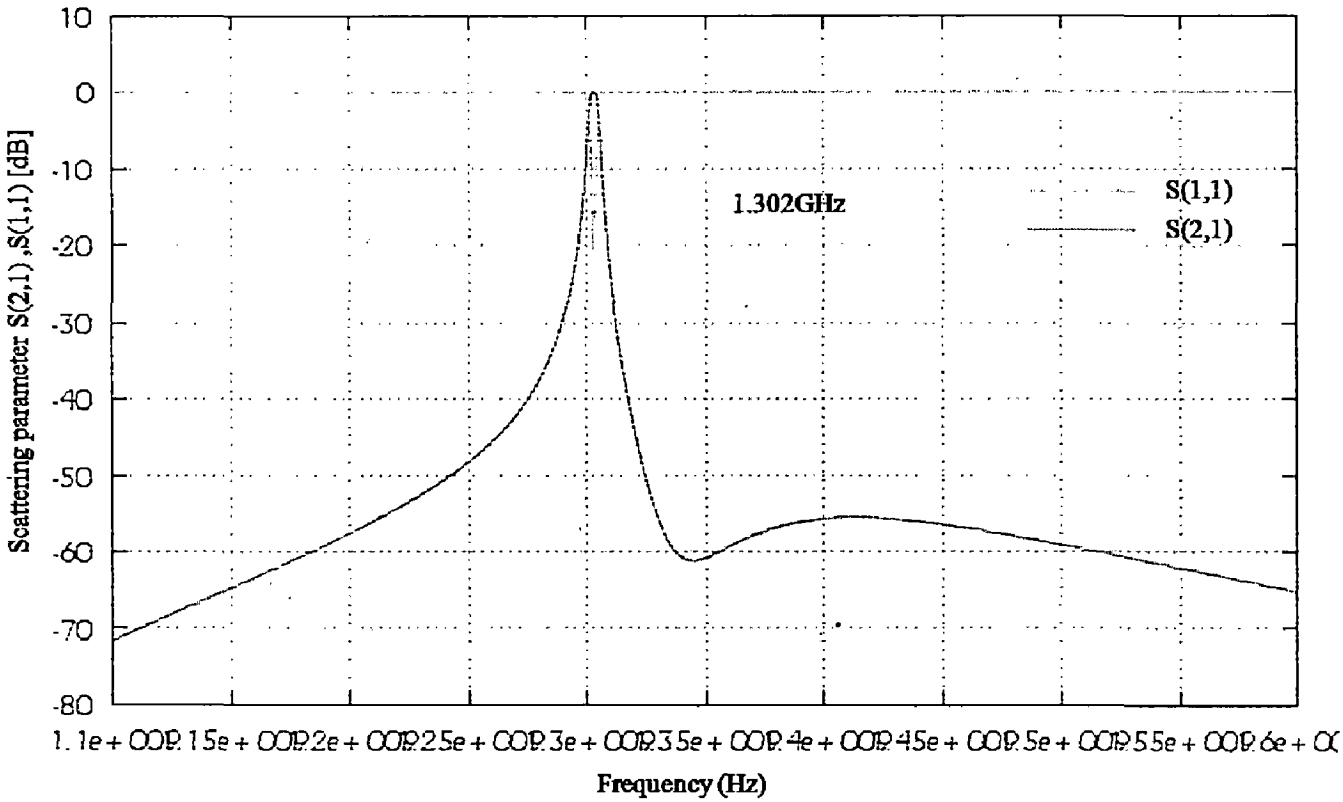
(g)


1.167GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

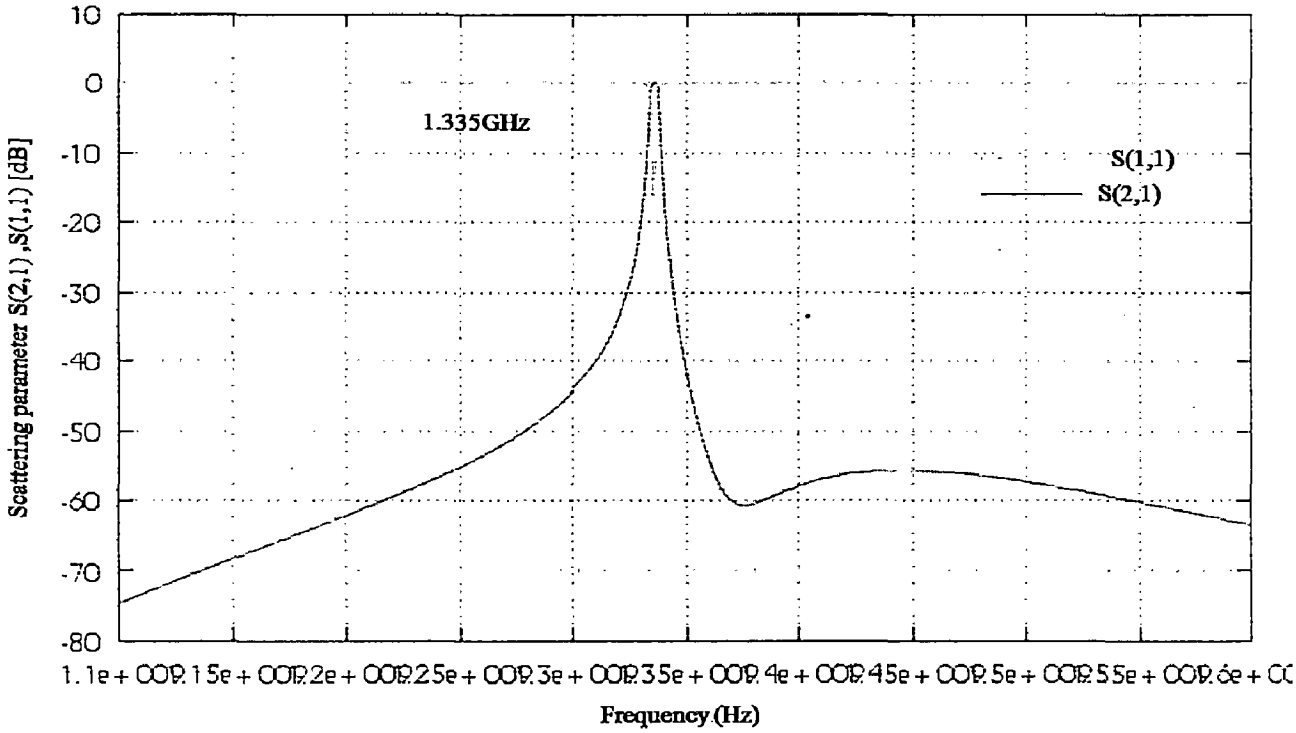
(h)


1.229GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
9e+008
1e+009
1.1e+009
1.2e+009
1.3e+009
1.4e+009
1.5e+

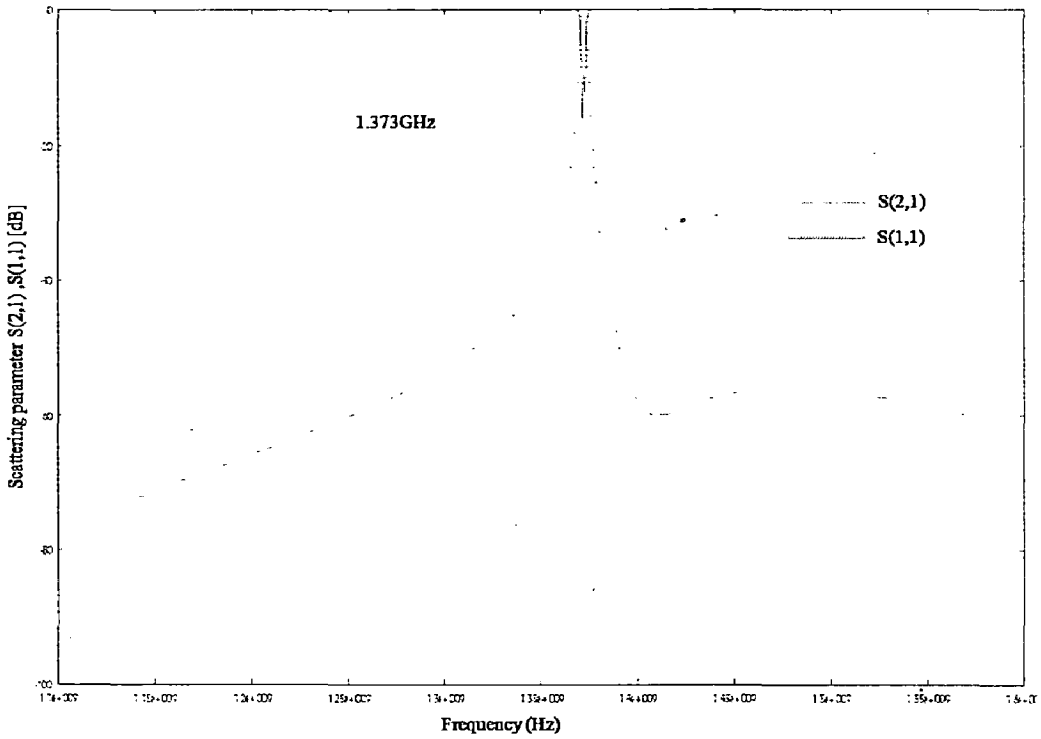
(i)


1.302GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80

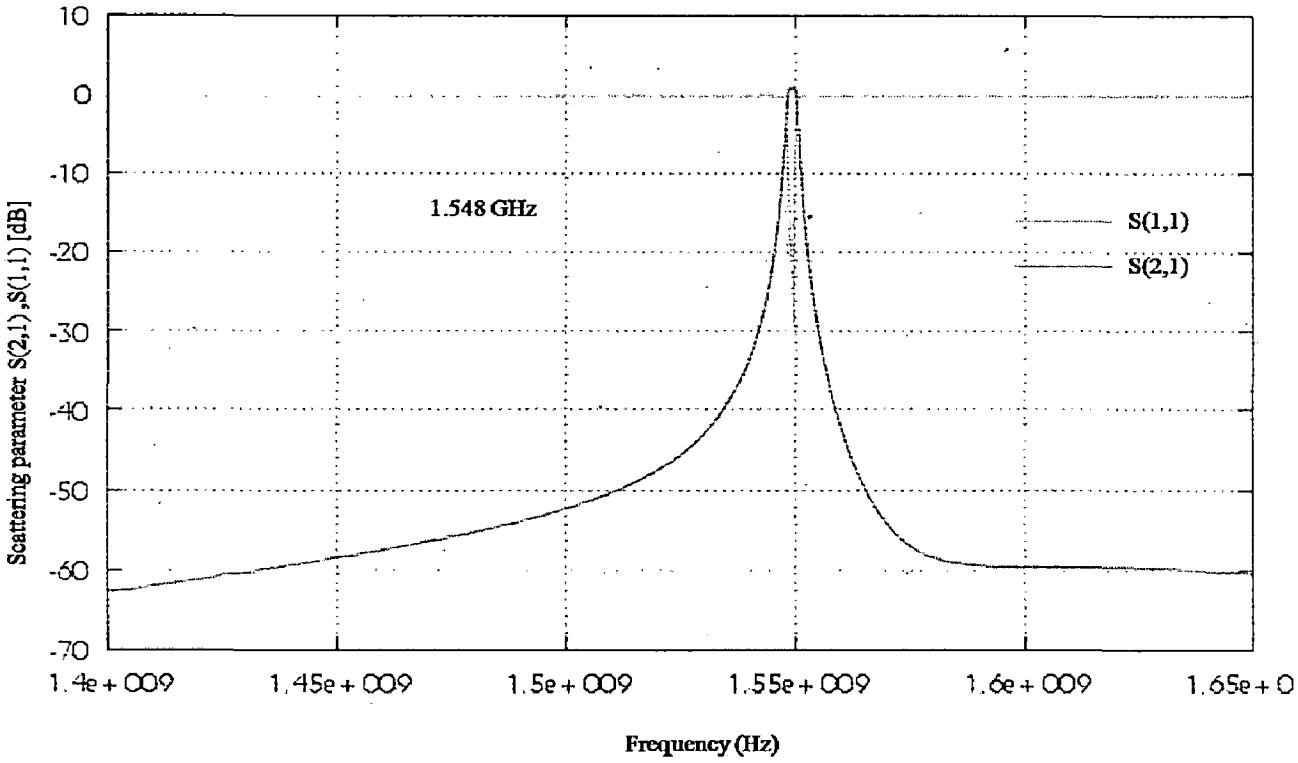
(j)


1.335GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

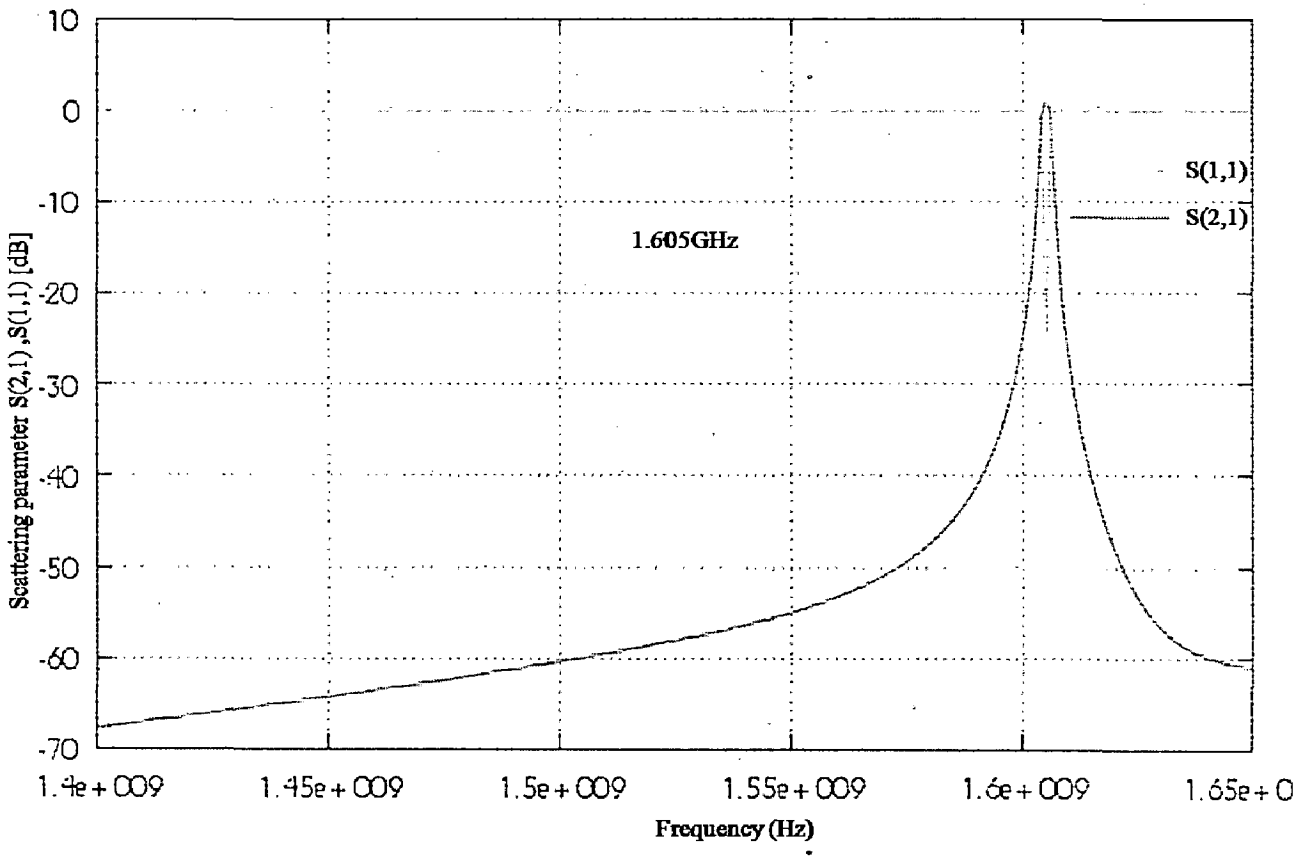
(k)


1.373GHz
S(2,1)
S(1,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

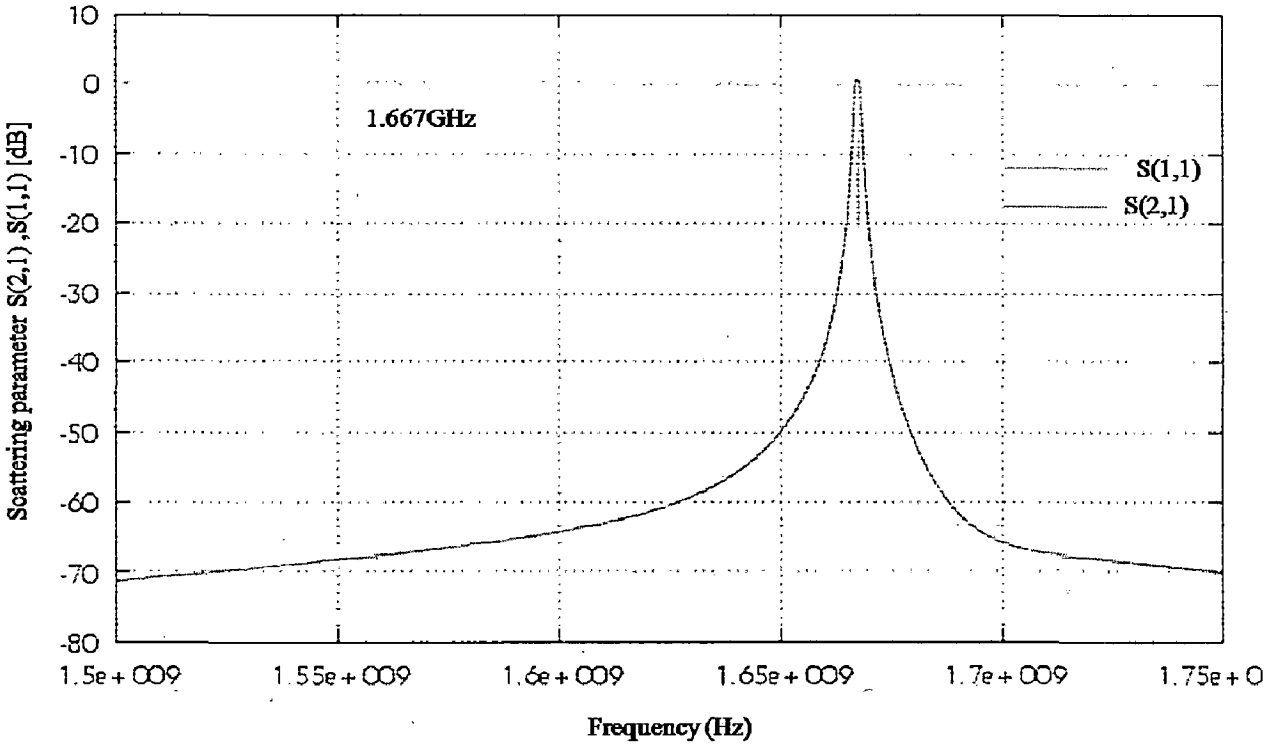
(l)


10
0
-10
-20
-30
-40
-50
-60
-70
1.548 GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1),S(1,1) [dB]
1.4e+009
1.45e+009
1.5e+009
1.55e+009
1.6e+009
1.65e+0
Frequency (Hz)

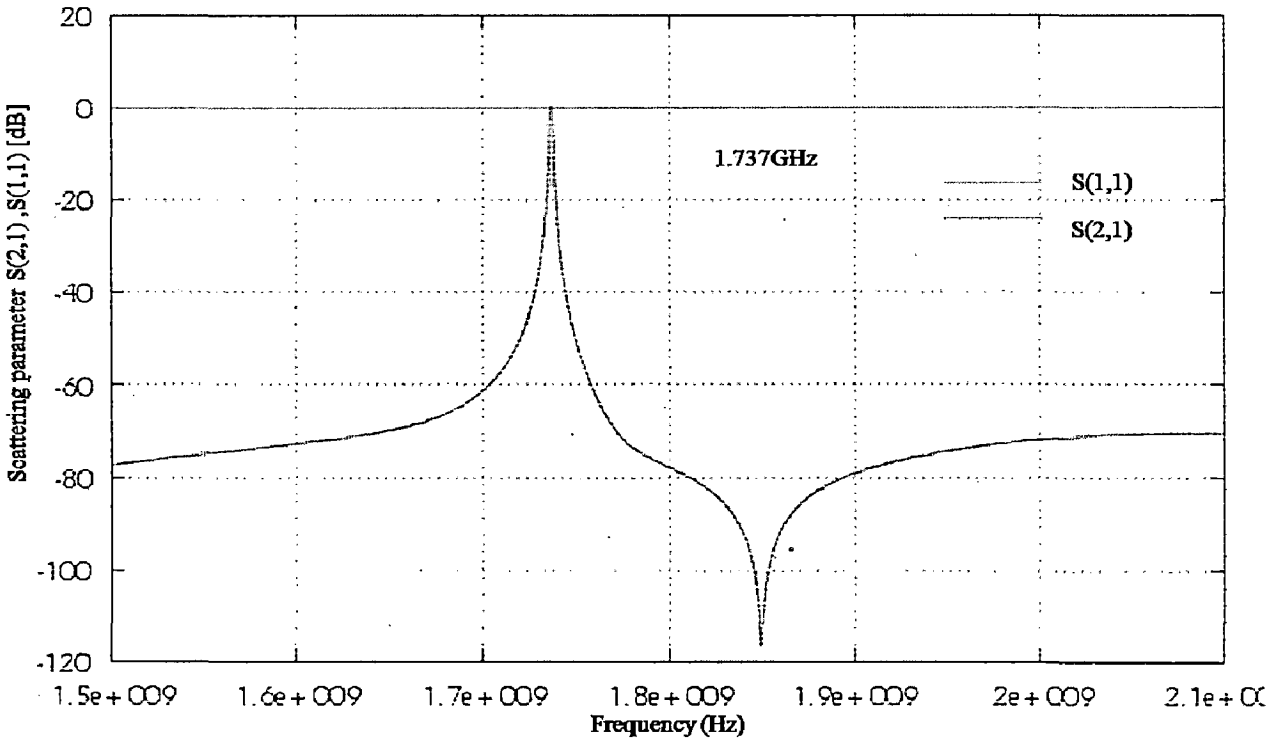
(m)


10
0
-10
-20
-30
-40
-50
-60
-70
1.605GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1),S(1,1) [dB]
1.4e+009
1.45e+009
1.5e+009
1.55e+009
1.6e+009
1.65e+0
Frequency (Hz)

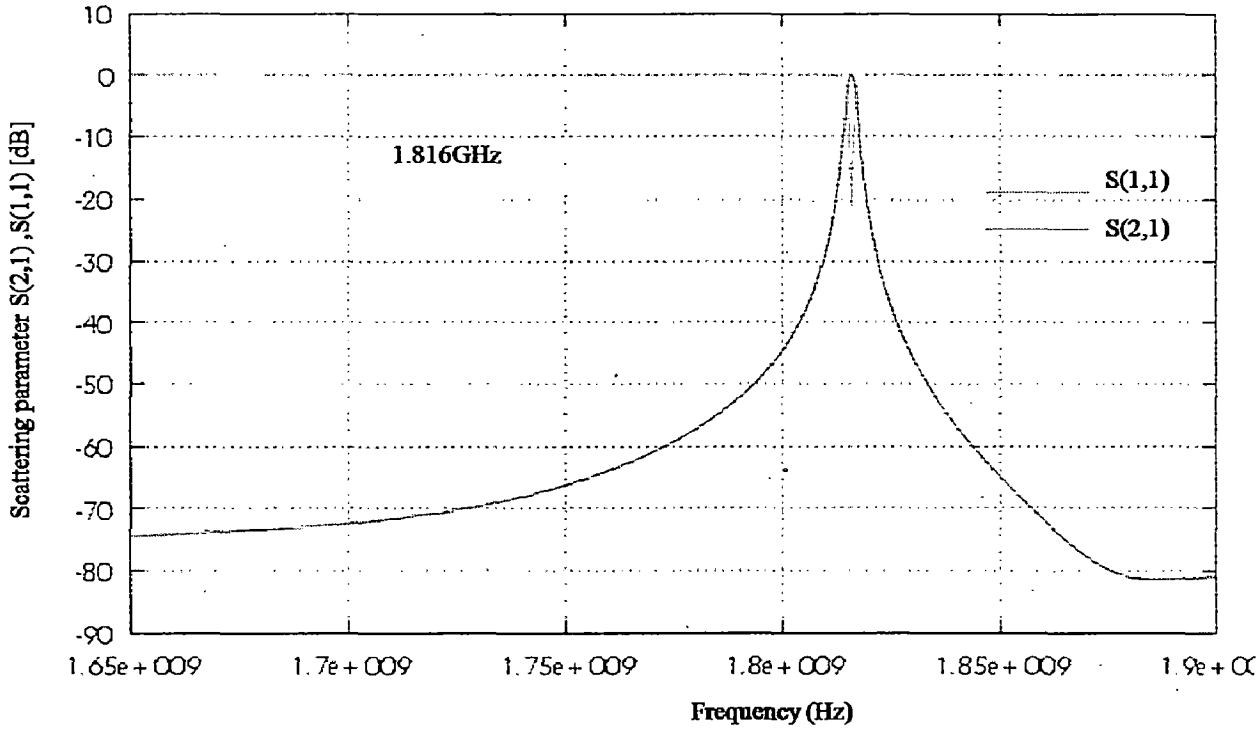
(n)


1.667GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
1.5e+009
1.55e+009
1.6e+009
1.65e+009
1.7e+009
1.75e+0

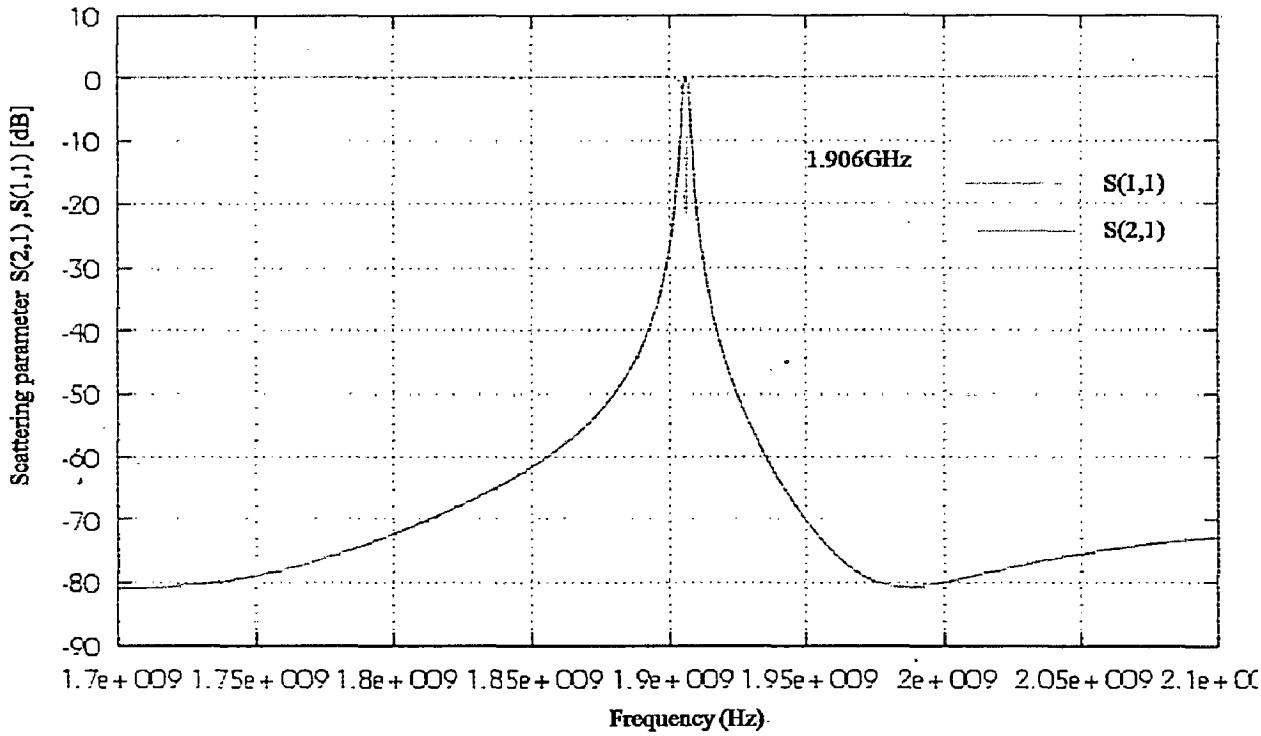
(o)


1.737GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
20
0
-20
-40
-60
-80
-100
-120
1.5e+009
1.6e+009
1.7e+009
1.8e+009
1.9e+009
2e+009
2.1e+0

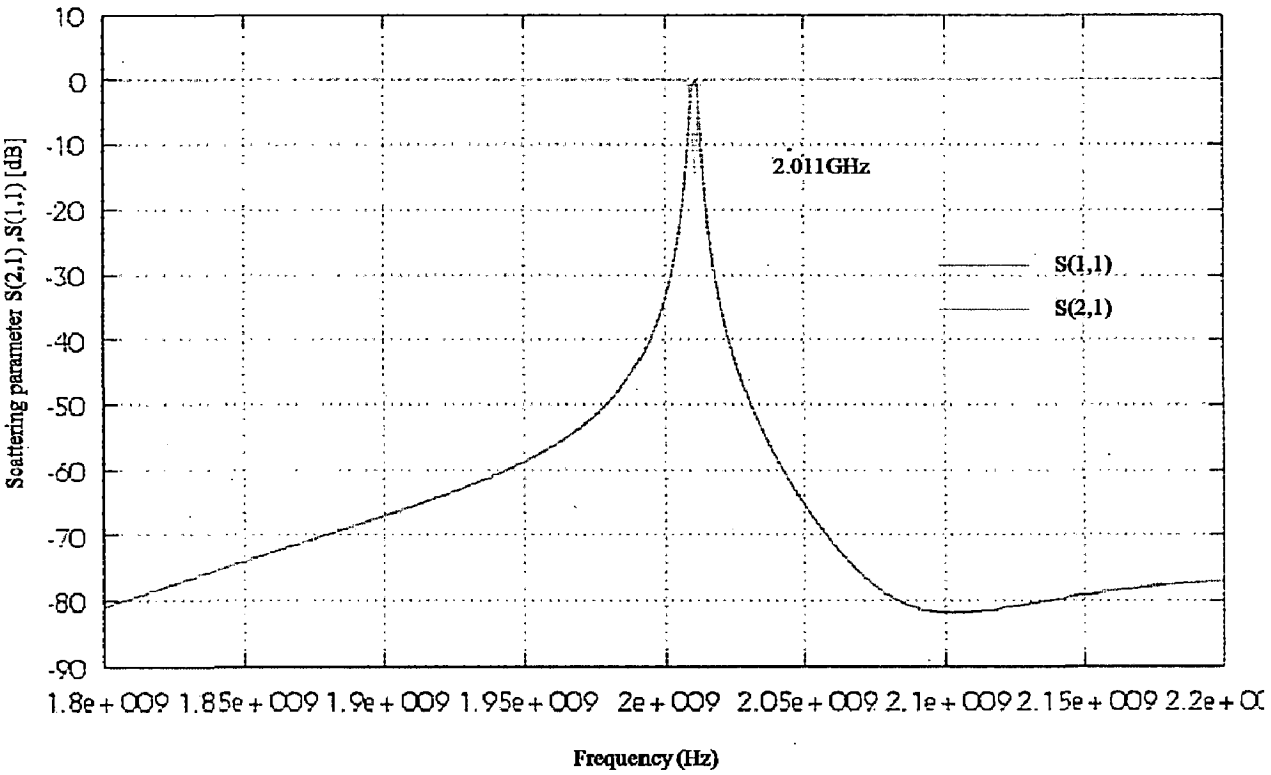
(p)


1.816GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

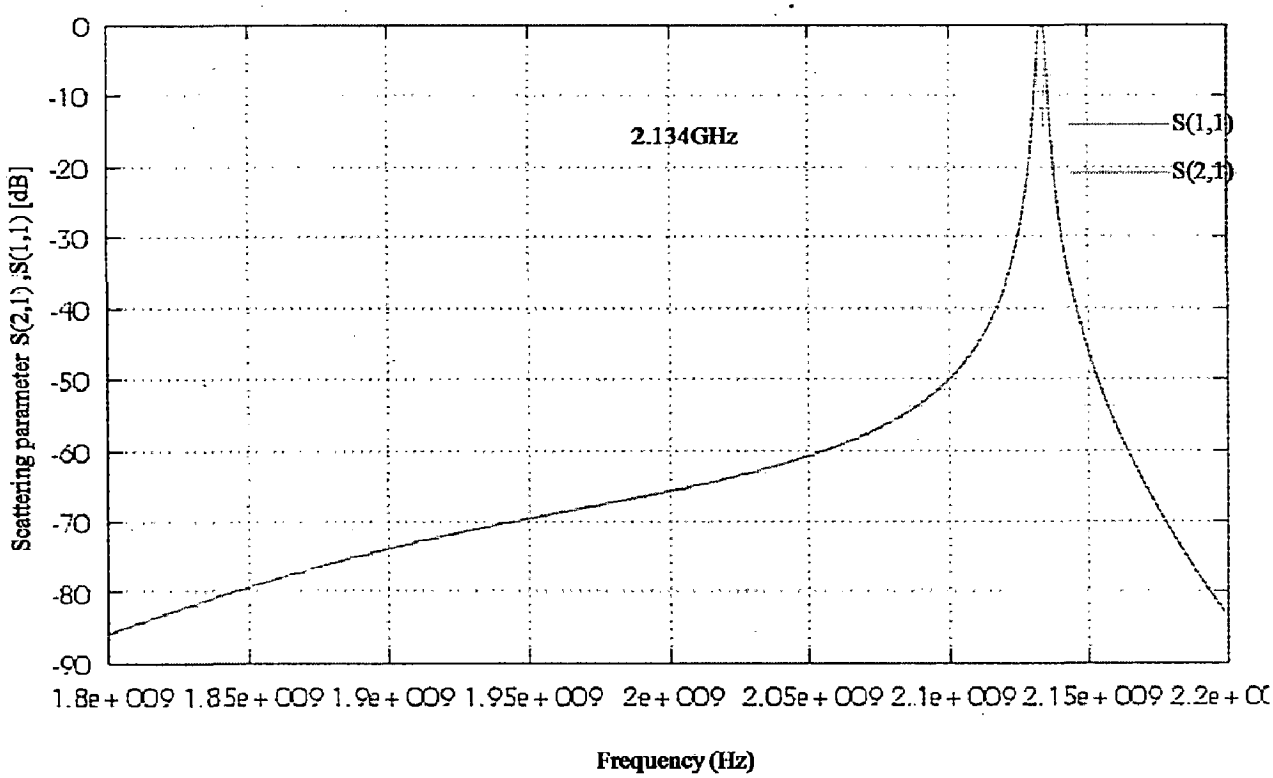
(q)


1.906GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)

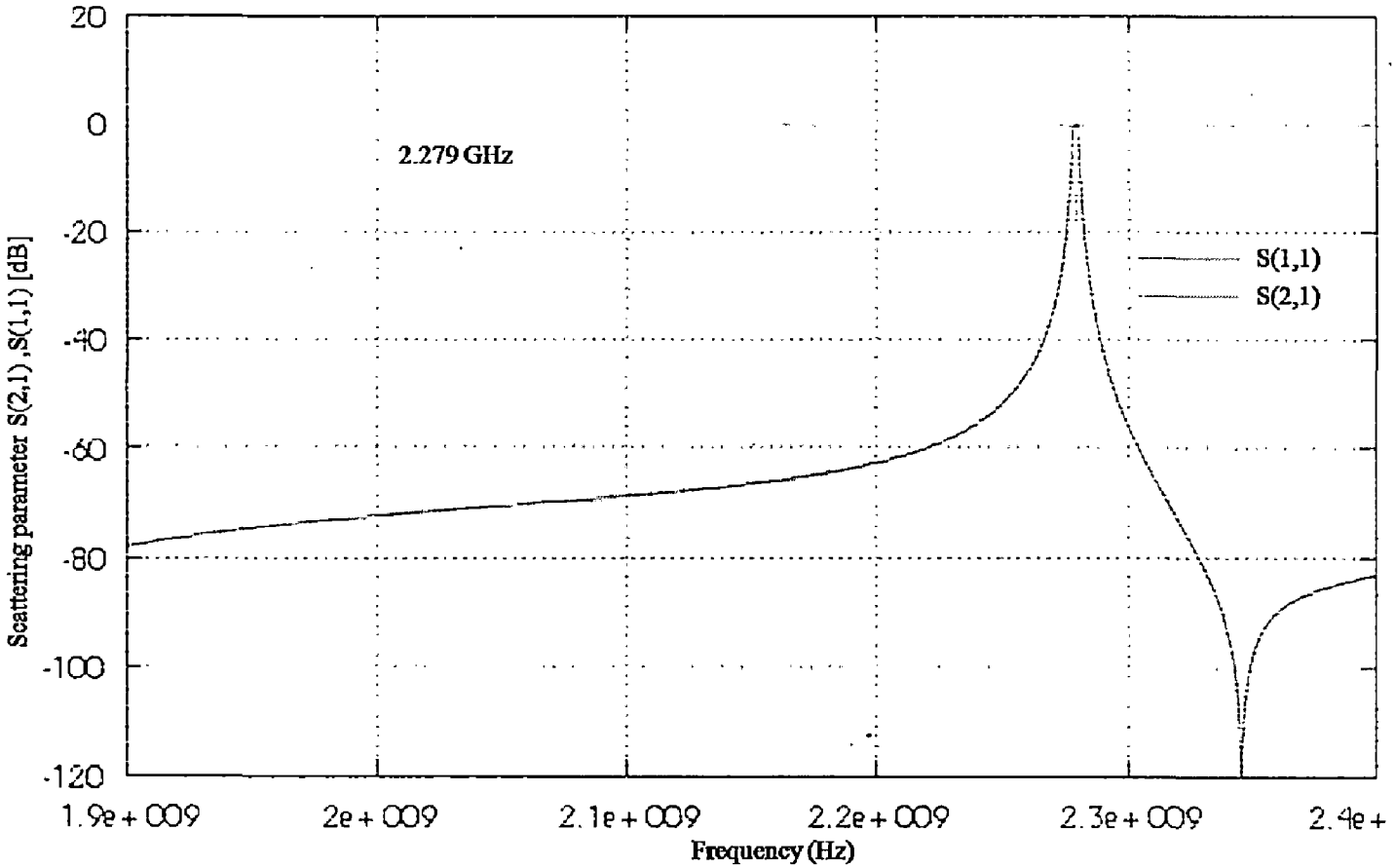
(r)


10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
Scattering parameter S(2,1) ,S(1,1) [dB]
2.011GHz
S(1,1)
S(2,1)
1.8e+009 1.85e+009 1.9e+009 1.95e+009 2e+009 2.05e+009 2.1e+009 2.15e+009 2.2e+0
Frequency (Hz)

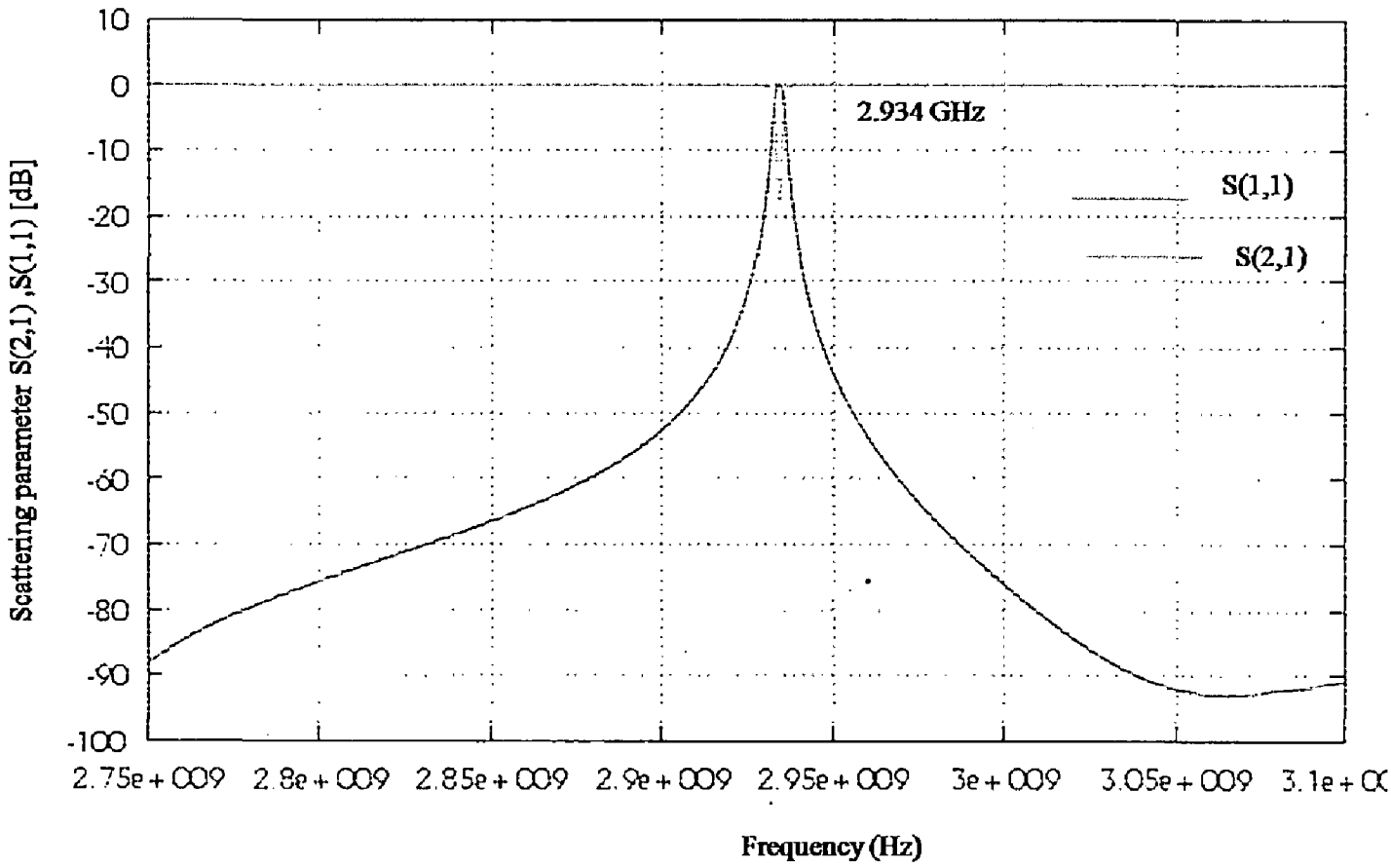
(s)


0
-10
-20
-30
-40
-50
-60
-70
-80
-90
Scattering parameter S(2,1) ,S(1,1) [dB]
2.134GHz
S(1,1)
S(2,1)
1.8e+009 1.85e+009 1.9e+009 1.95e+009 2e+009 2.05e+009 2.1e+009 2.15e+009 2.2e+0
Frequency (Hz)

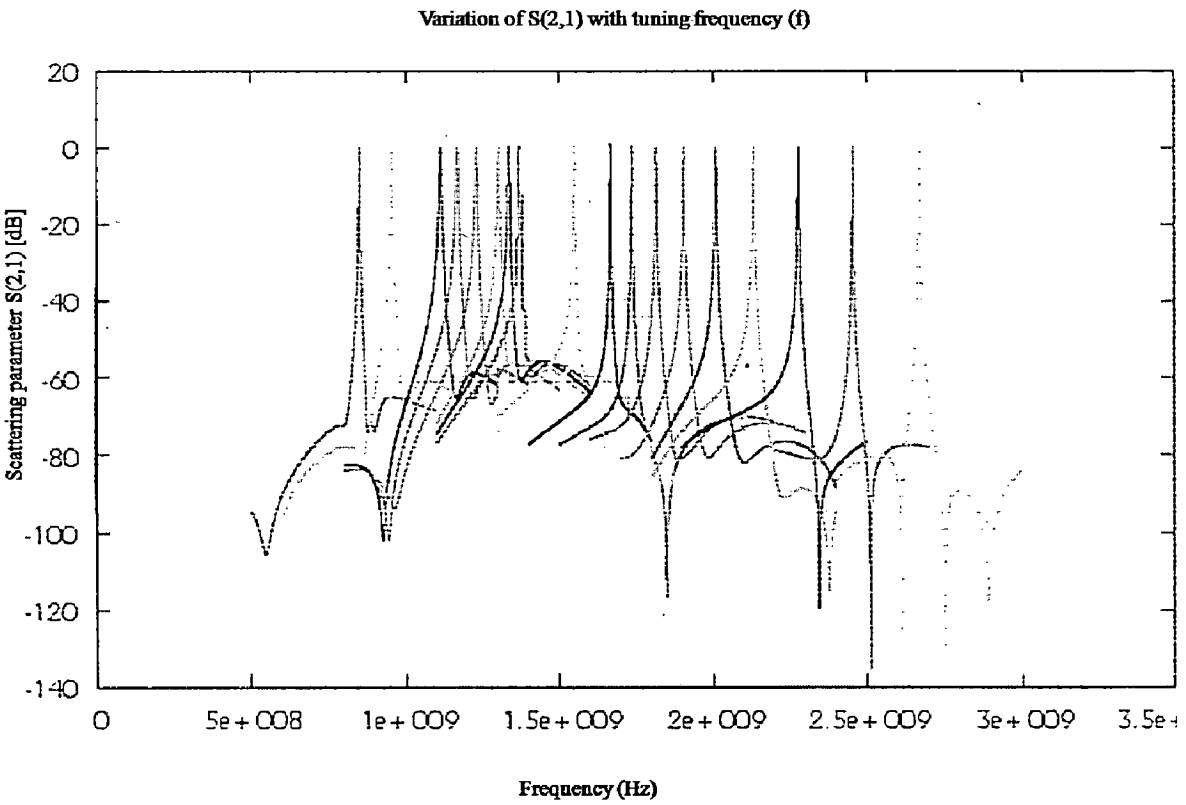
(t)


2.279 GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
20
0
-20
-40
-60
-80
-100
-120
1.9e+009
2e+009
2.1e+009
2.2e+009
2.3e+009
2.4e+

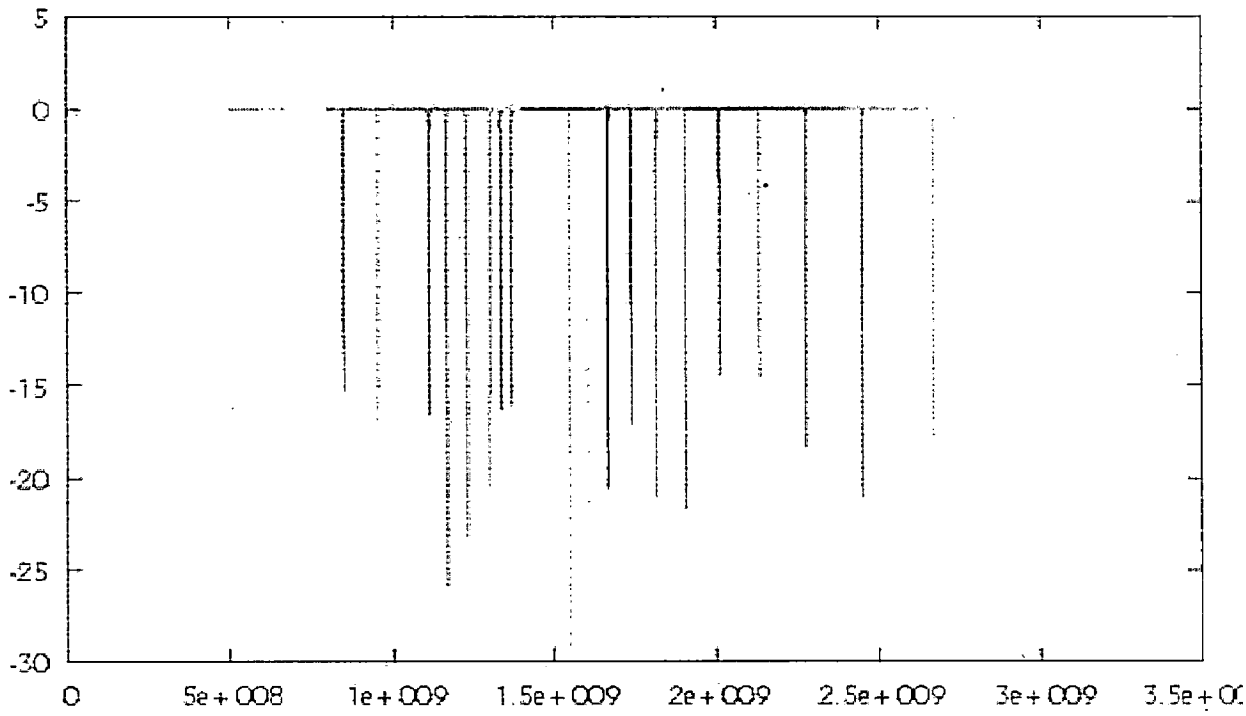
(u)


2.934 GHz
S(1,1)
S(2,1)
Scattering parameter S(2,1) ,S(1,1) [dB]
Frequency (Hz)
10
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
2.75e+009
2.8e+009
2.85e+009
2.9e+009
2.95e+009
3e+009
3.05e+009

(v)



(w)



(x)

Fig.3.13: Simulation Results With Ideal Tunable Capacitors

Layout Of The Filter Using The Parasitic Effects Of The Varactor

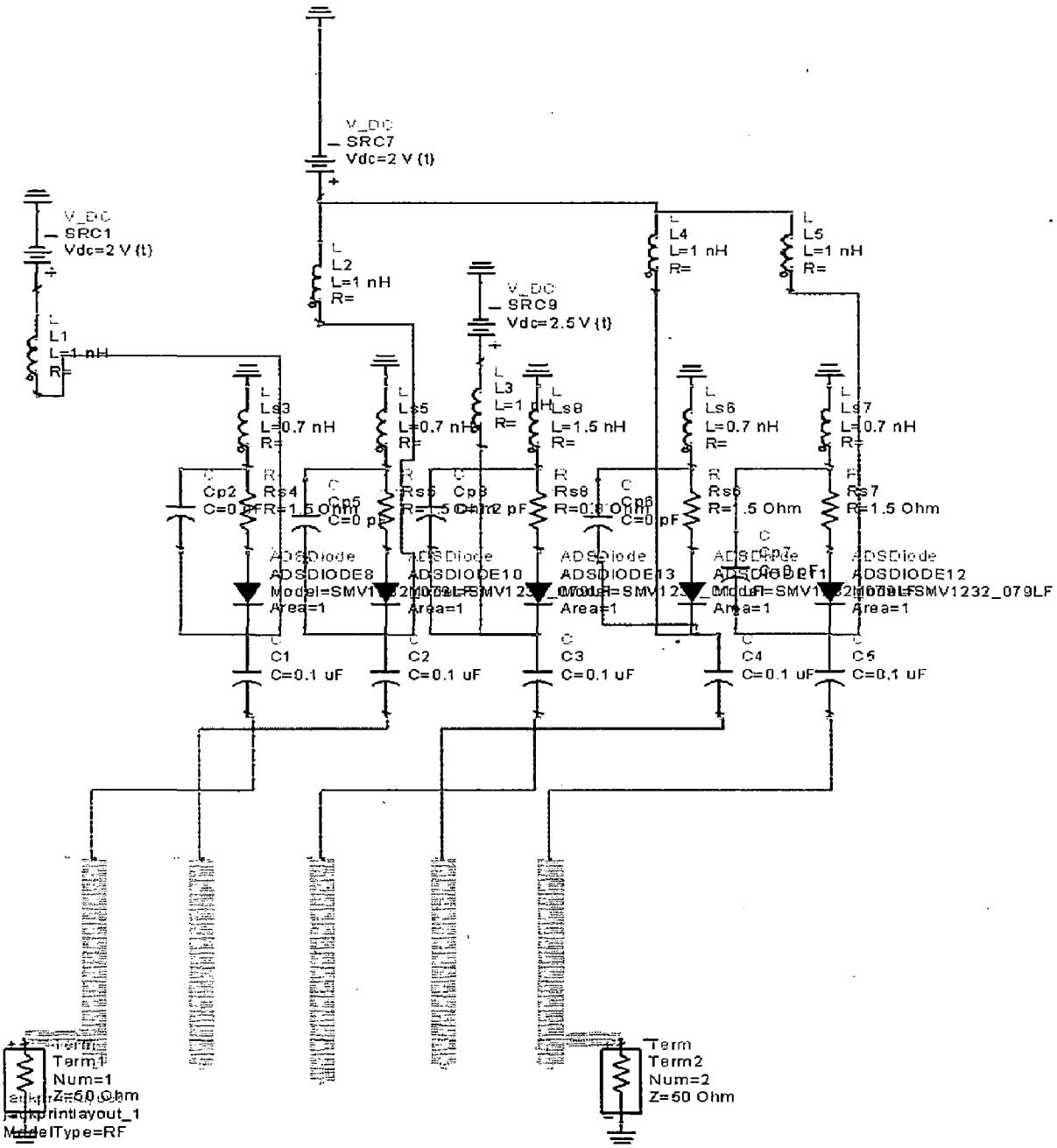
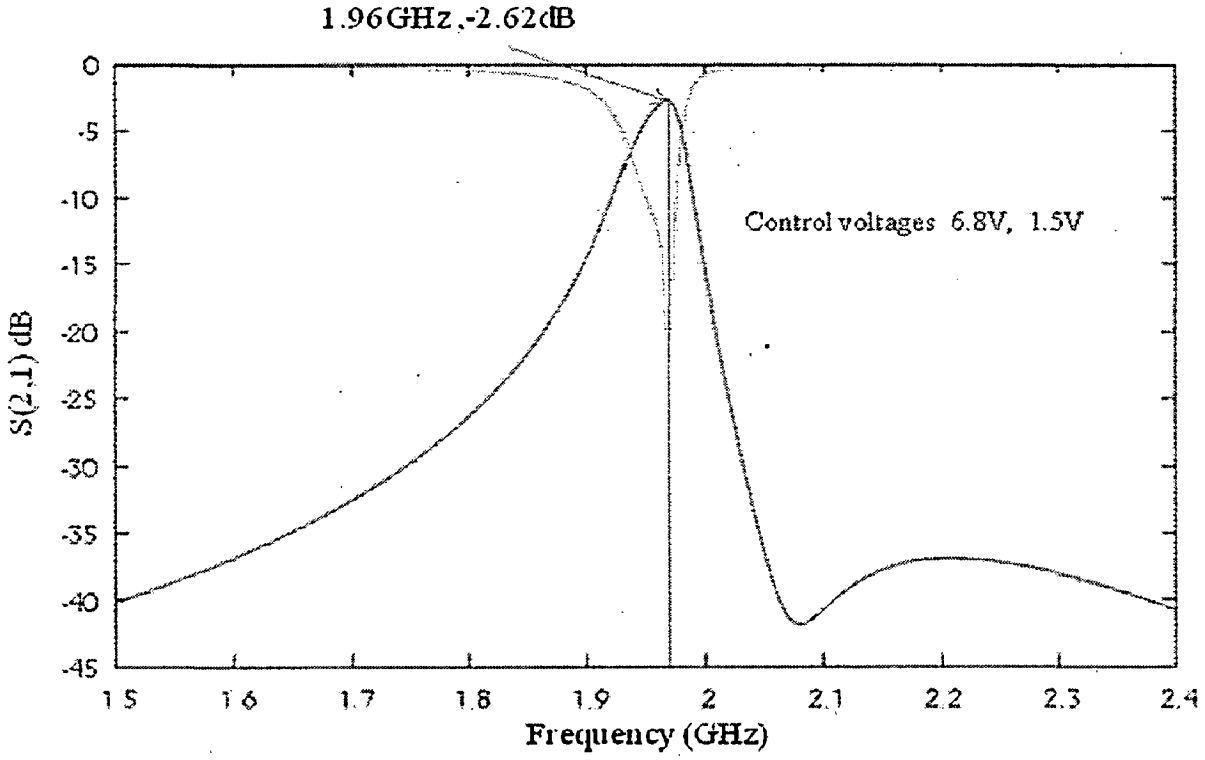
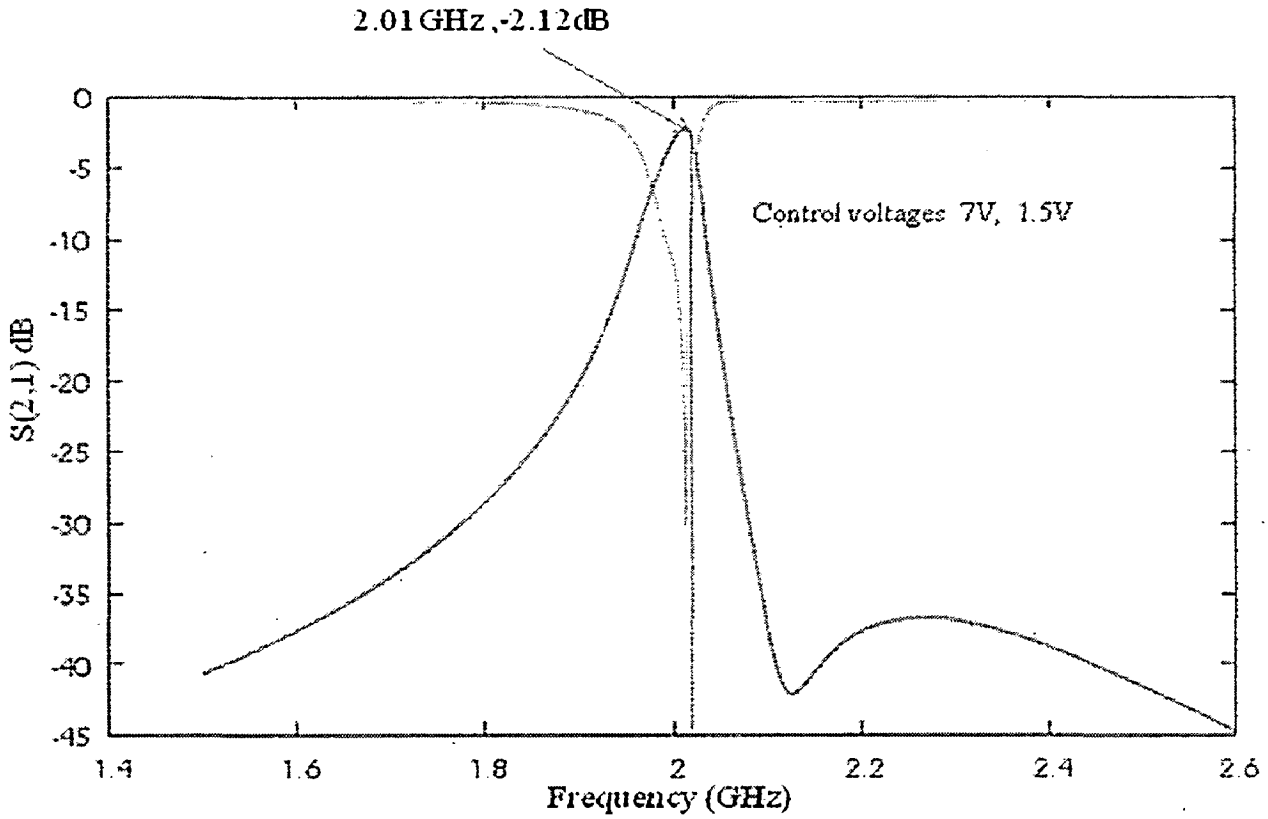


Fig 3.14: Layout of the filter using the spice model of the varactor

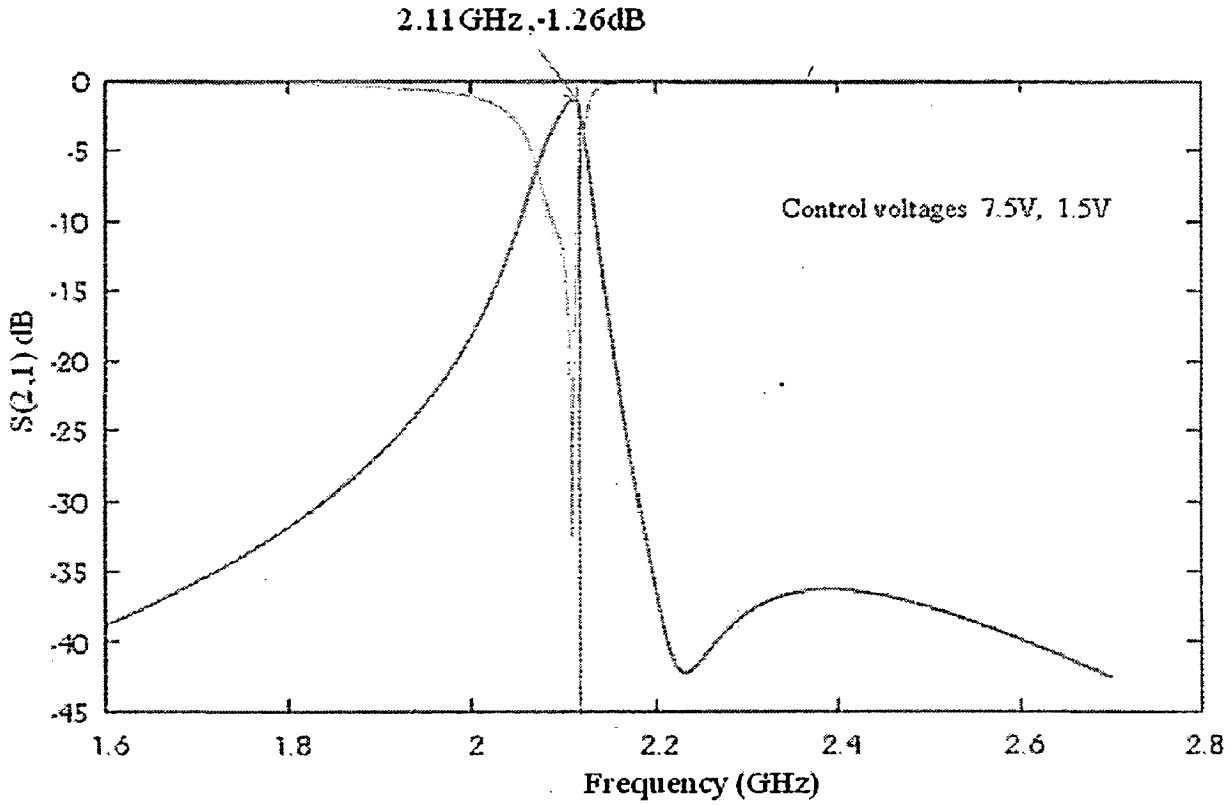
Simulation Results Including The Parasitic Effects Of The Varactors:



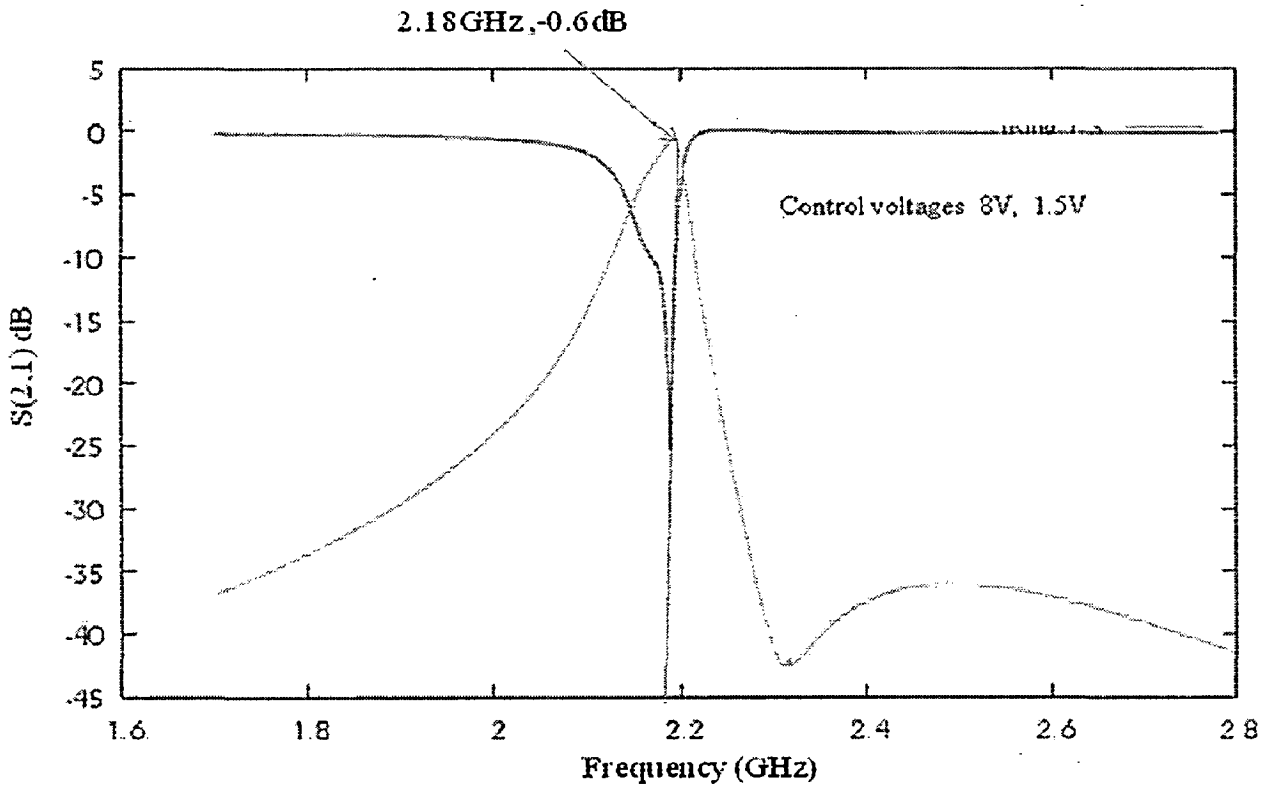
(a)



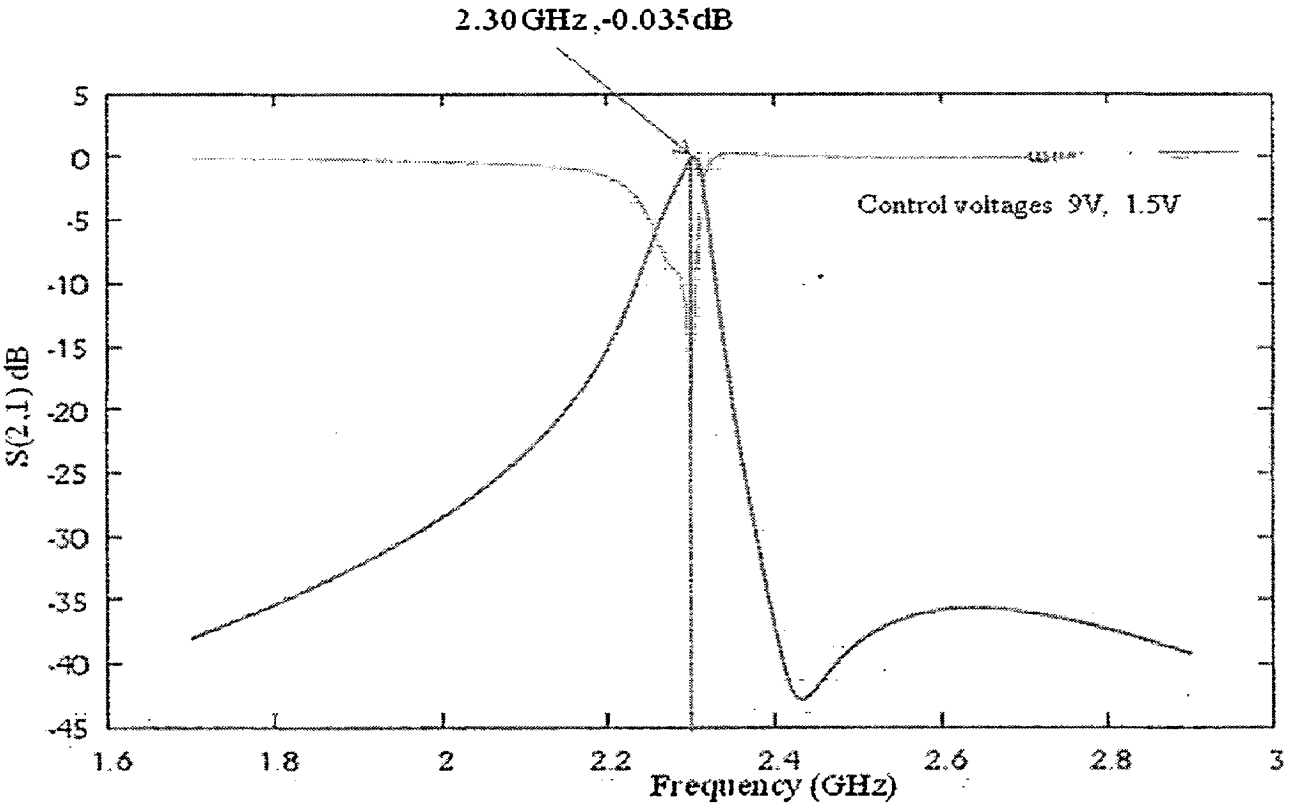
(b)


2.11GHz,-1.26dB
Control voltages 7.5V, 1.5V
S(2,1) dB
Frequency (GHz)
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.6
1.8
2
2.2
2.4
2.6
2.8

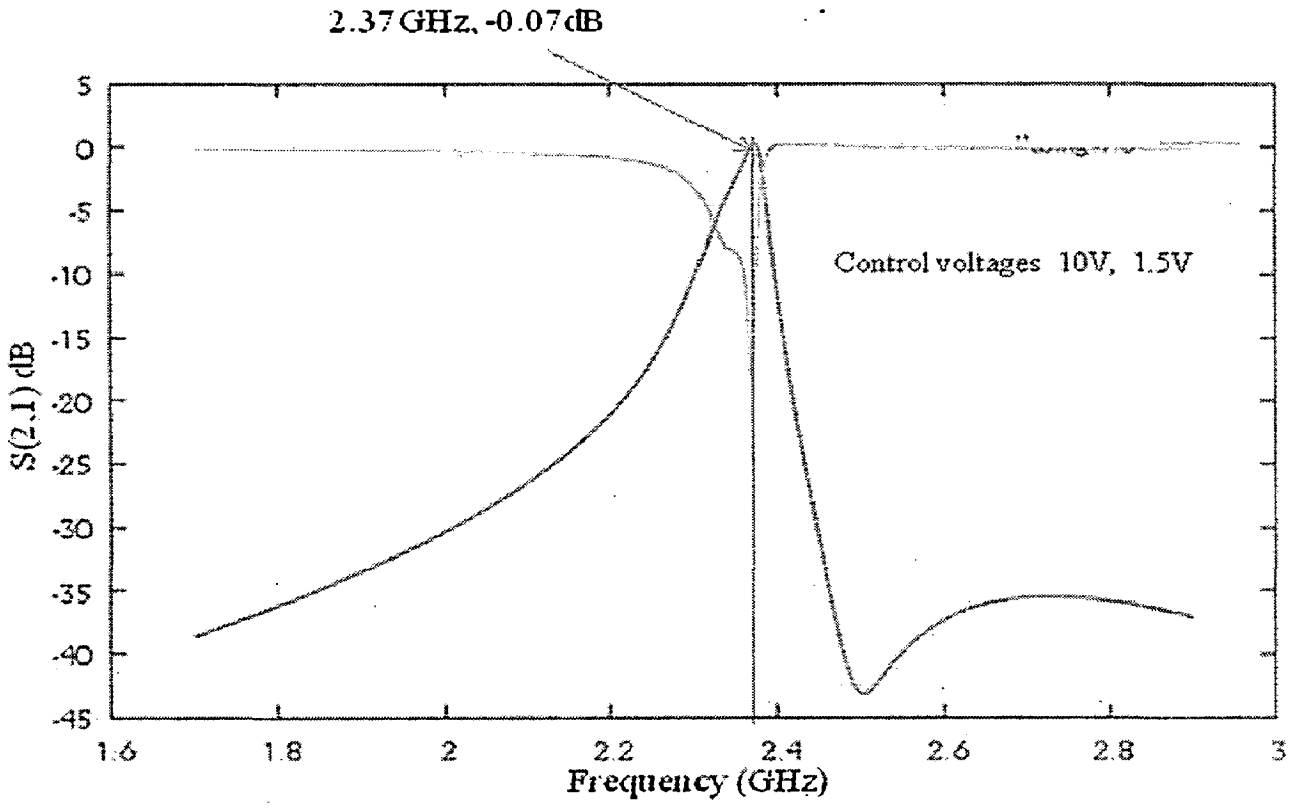
(c)


2.18GHz,-0.6dB
Control voltages 8V, 1.5V
S(2,1) dB
Frequency (GHz)
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.6
1.8
2
2.2
2.4
2.6
2.8

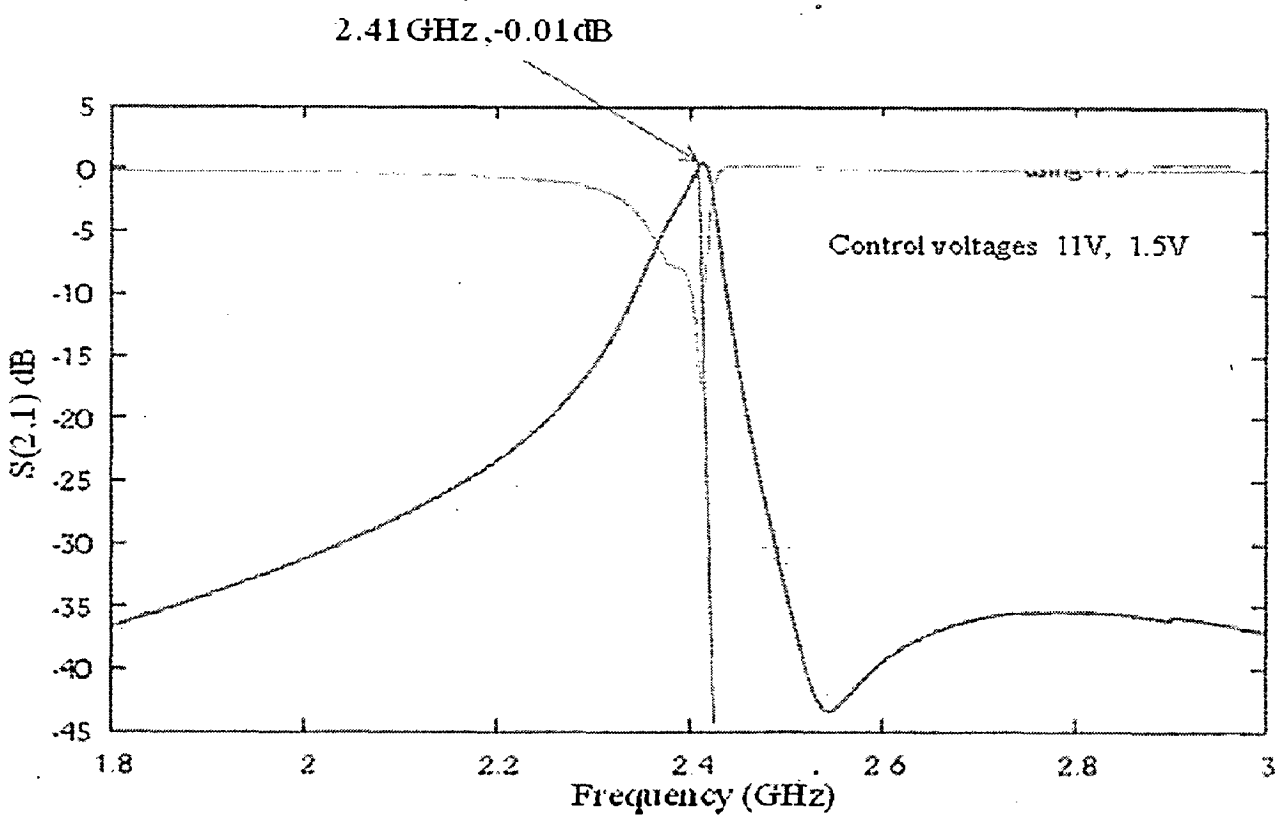
(d)


2.30 GHz, -0.035 dB
Control voltages 9V, 1.5V
S(2,1) dB
Frequency (GHz)
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.6
1.8
2
2.2
2.4
2.6
2.8
3

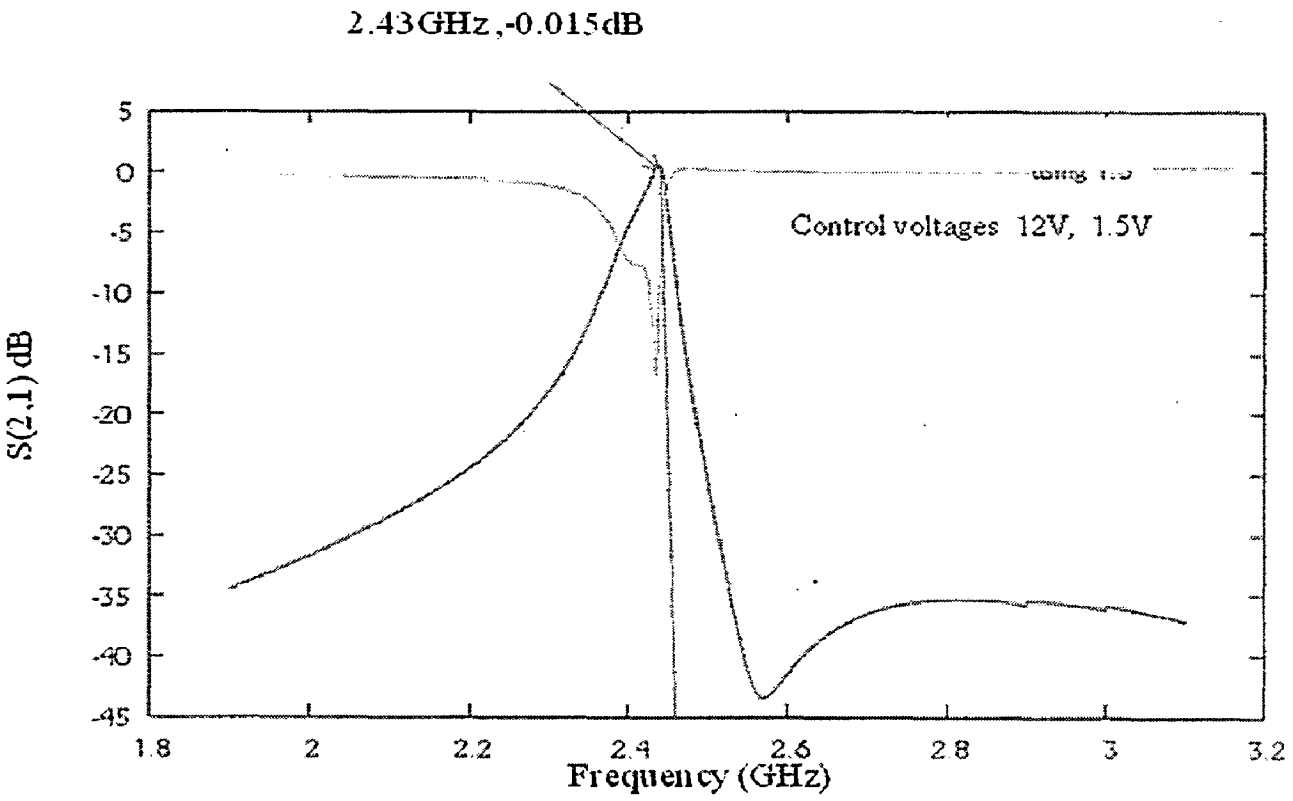
(e)


2.37 GHz, -0.07 dB
Control voltages 10V, 1.5V
S(2,1) dB
Frequency (GHz)
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.6
1.8
2
2.2
2.4
2.6
2.8
3

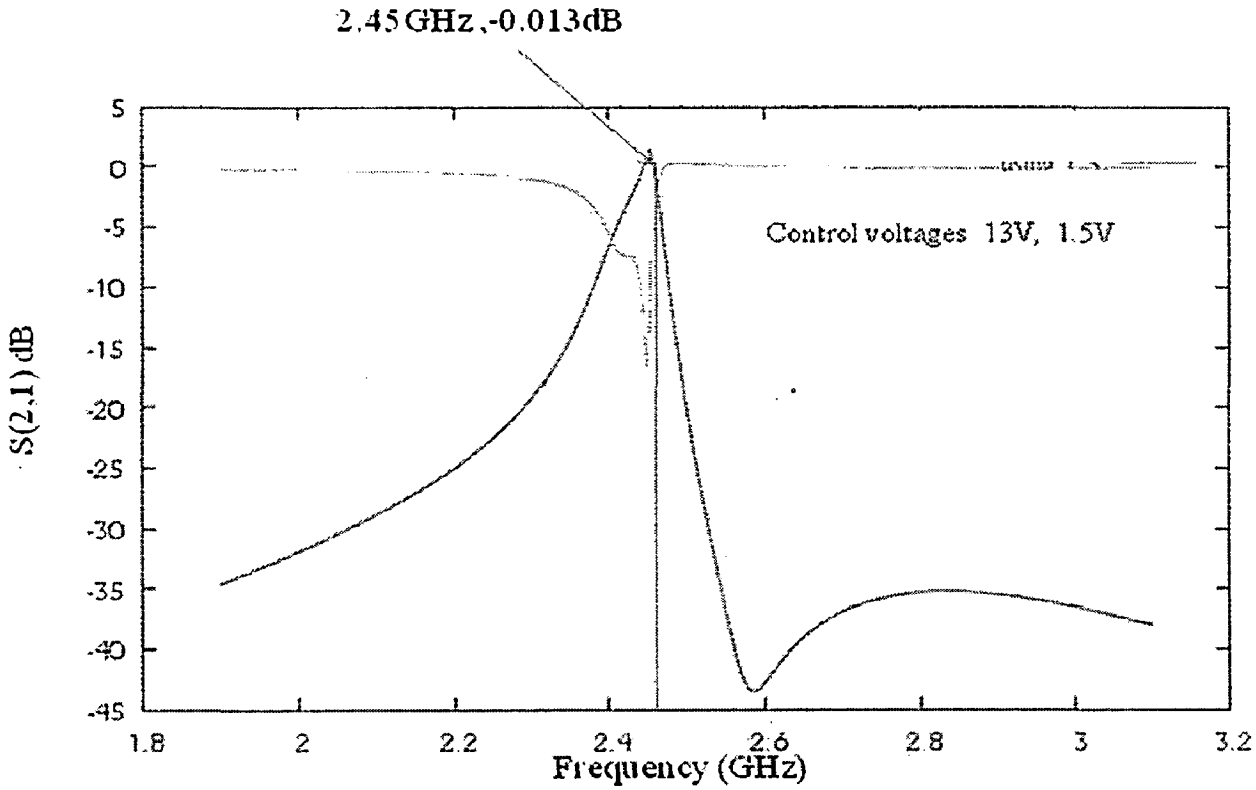
(f)


2.41GHz,-0.01dB
Control voltages 11V, 1.5V
S(2,1) dB
Frequency (GHz)
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.8
2
2.2
2.4
2.6
2.8
3

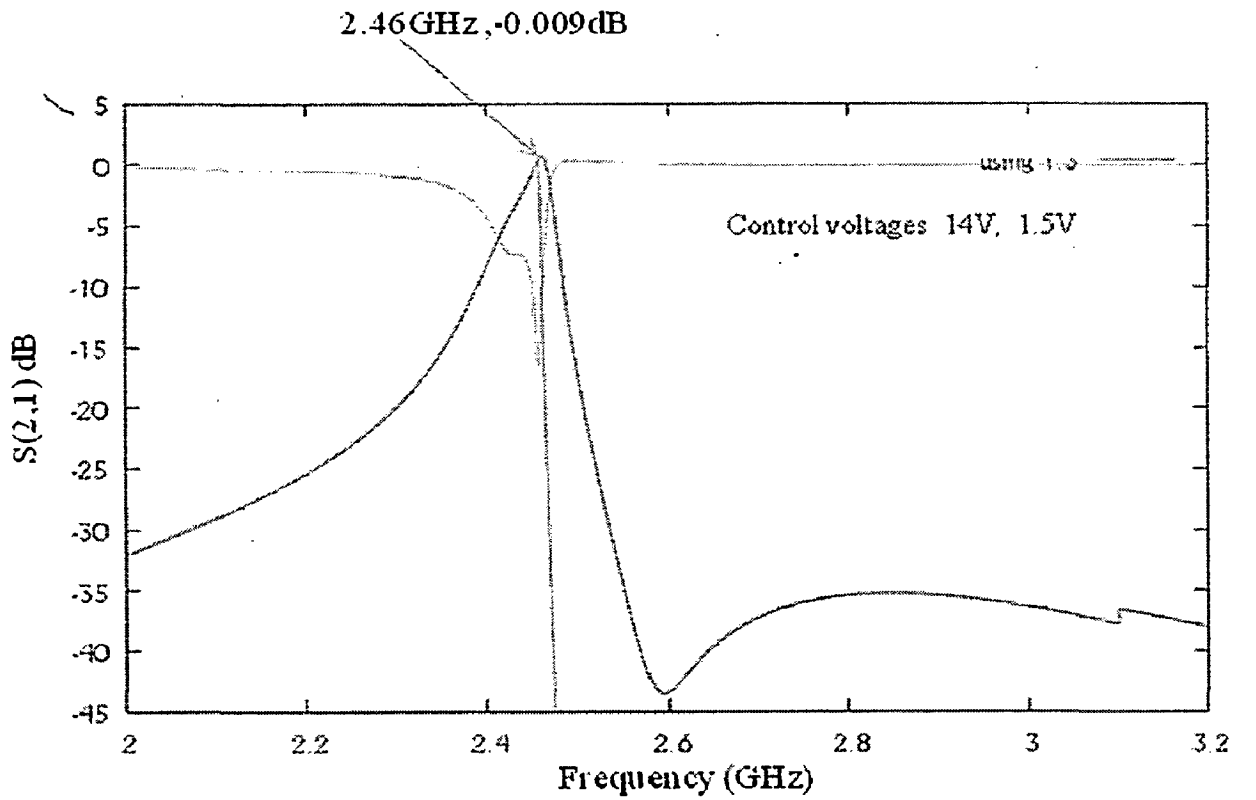
(g)


2.43GHz,-0.015dB
Control voltages 12V, 1.5V
S(2,1) dB
Frequency (GHz)
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
1.8
2
2.2
2.4
2.6
2.8
3
3.2

(h)



(i)



(j)

Fig. 3.15: Simulation Results Including The Parasitic Effects Of The Varactors

3.6. Conclusion

In this chapter step by step design procedure for the tunable combline filter is presented starting from the specifications. The designed filter is simulated using ADS Momentum and the simulation results with ideal tunable capacitors is presented. The varactor spicemodel is introduced and the filter is simulated again by taking the package effects of the varactor.

Chapter 4
CAD For Designing Tunable Combline Filter

4.1. Introduction

This chapter deals with the computer aided designing of tunable combline filter. For designing the tunable combline filter a MATLAB [26] code was prepared .The specifications of the filter are provided as the input to the CAD.

The specifications required for the calculation of the order of the filter

1. Equi ripple bandwidth
2. Pass band Ripple
3. Stop Band Rejection
4. Stop Band frequency

4.2. MATLAB Code and Results

The first program calculates the order of the filter from the given specifications and is written as follows:

```
function varargout = Mygui(varargin)
% MYGUI M-file for Mygui.fig
%      MYGUI, by itself, creates a new MYGUI or raises the existing
%      singleton*.
%
%      H = MYGUI returns the handle to a new MYGUI or the handle to
%      the existing singleton*.
%
%      MYGUI('CALLBACK', hObject,eventData,handles,...) calls the local
%      function named CALLBACK in MYGUI.M with the given input arguments.
%
%      MYGUI('Property', 'Value',...) creates a new MYGUI or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Mygui_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
```

```
%       stop.  All inputs are passed to Mygui_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help Mygui
% Last Modified by GUIDE v2.5 09-Jul-2010 18:12:00
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Mygui_OpeningFcn, ...
                   'gui_OutputFcn',  @Mygui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Mygui is made visible.
function Mygui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% varargin   command line arguments to Mygui (see VARARGIN)
% Choose default command line output for Mygui
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes Mygui wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = Mygui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%for calculating order of the filter
f1 = get(handles.edit1,'string');
f1 = str2num(f1);
f2 = get(handles.edit2,'string');
f2 = str2num(f2);
ripple_bandwidth = get(handles.edit3,'string');
ripple_bandwidth = str2num(ripple_bandwidth);
Lar = get(handles.edit4,'string');
Lar = str2num(Lar);
Las = get(handles.edit5,'string');
Las = str2num(Las);
```

```
f0=sqrt(f1*f2);
fprintf('%s%4.4f\n','f0 = ',f0)
deltaW=(ripple_bandwidth/f0)/1000;
w=f0+(80/1000);
w0=f0;
w1=((w/w0)-(w0/w))/deltaW;
c=(((10.^(0.1*Las))-1)/((10.^(0.1*Lar))-1));
c=0.1*Las;
c=10.^c;
c=c-1;
d=0.1*Lar;
d=10.^d;
d=d-1;
c=sqrt(c/d);
e=acosh(c);
f=acosh(w1);
n=e/f;
fprintf('%s%4.4f\n','n = ',n)
set(handles.text10,'string',num2str(f0));
set(handles.text6,'string',num2str(n));
function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double
% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as a
double
% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a
double
% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

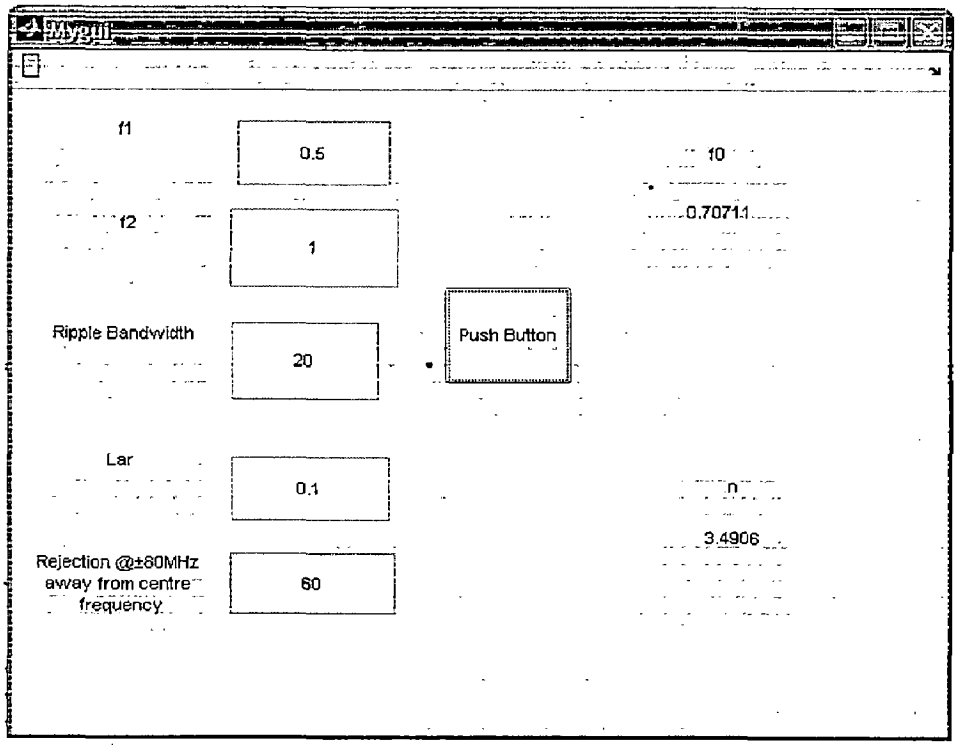


Fig. 4.1: GUI for Calculating the Filter Order, n

MATLAB code for calculating filter coefficients, coupling coefficients and quality factor

```
function varargout = Mygui1(varargin)
% MYGUI1 M-file for Mygui1.fig
%      MYGUI1, by itself, creates a new MYGUI1 or raises the existing
%      singleton*.
%
%      H = MYGUI1 returns the handle to a new MYGUI1 or the handle to
%      the existing singleton*.
%
%      MYGUI1('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in MYGUI1.M with the given input arguments.
%
%      MYGUI1('Property','Value',...) creates a new MYGUI1 or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Mygui1_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Mygui1_OpeningFcn via varargin.
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Mygui1_OpeningFcn, ...
                   'gui_OutputFcn',  @Mygui1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Myguil is made visible.
function Myguil_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for Myguil
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes Myguil wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = Myguil_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function edit1_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
n1 = get(handles.edit1,'string');
```

```
n1 = str2num(n1);
f1 = get(handles.edit3,'string');
f1 = str2num(f1);
f2 = get(handles.edit4,'string');
f2 = str2num(f2);
ripple_bandwidth= get(handles.edit2,'string');
ripple_bandwidth = str2num(ripple_bandwidth);
%for calculating beta and gamma
%actual order(n1) will be given by the user which will be greater than n
f0=sqrt(f1*f2);
fprintf('%s%4.4f\n','f0 = ',f0)
Lar=0.1;
deltaW=(ripple_bandwidth/f0)/1000;
beta=log(coth(Lar/17.37));
gamma=sinh(beta/(2*n1));
fprintf('%s%4.4f\n','beta = ',beta)
fprintf('%s%4.4f\n','gamma = ',gamma)
%for calculating the filter coefficients
g0=1.0;
g(1)=(2*sin(pi/(2*n1)))/gamma;
fprintf('%s%4.4f\n','g(1) = ',g(1))
for k=2:n1
    l=(4*sin((2*k-1)*pi/(2*n1))*sin((2*k-3)*pi/(2*n1)));
    m=gamma^2;
    o=sin(((k-1)*pi)/n1)^2;
    m=m+o;
    p=l/m;
    g(k)=p/g(k-1);
end
r=rem(n1,2);
if(r==0)
```

```
    g(n1+1)=(coth(beta/4))^2;
else
   g(n1+1)=1.0;
end
%for calculating the coupling coefficients
coupling_coefficient_1=deltaW/sqrt(g(1)*g(2));
fprintf('%s%4.4f\n','M1 = ',coupling_coefficient_1)
coupling_coefficient_2=deltaW/sqrt(g(2)*g(3));
fprintf('%s%4.4f\n','M2 = ',coupling_coefficient_2)
coupling_coefficient_3=deltaW/sqrt(g(3)*g(4));
fprintf('%s%4.4f\n','M3 = ',coupling_coefficient_3)
coupling_coefficient_4=deltaW/sqrt(g(4)*g(5));
fprintf('%s%4.4f\n','M4 = ',coupling_coefficient_4)
%for calculating quality factor
Q=(g0*g(1))/deltaW;
fprintf('%s%4.4f\n','Q = ',Q)
set(handles.text3,'string',num2str(beta));
set(handles.text5,'string',num2str(gamma));
set(handles.text22,'string',num2str(g(1)));
set(handles.text24,'string',num2str(g(2)));
set(handles.text26,'string',num2str(g(3)));
set(handles.text28,'string',num2str(g(4)));
set(handles.text30,'string',num2str(g(5)));
set(handles.text32,'string',num2str(g(6)));
set(handles.text16,'string',num2str(coupling_coefficient_1));
set(handles.text17,'string',num2str(coupling_coefficient_2));
set(handles.text18,'string',num2str(coupling_coefficient_3));
set(handles.text19,'string',num2str(coupling_coefficient_4));
set(handles.text20,'string',num2str(Q));
function edit2_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
```

```
function edit2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit3_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit4_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end
```

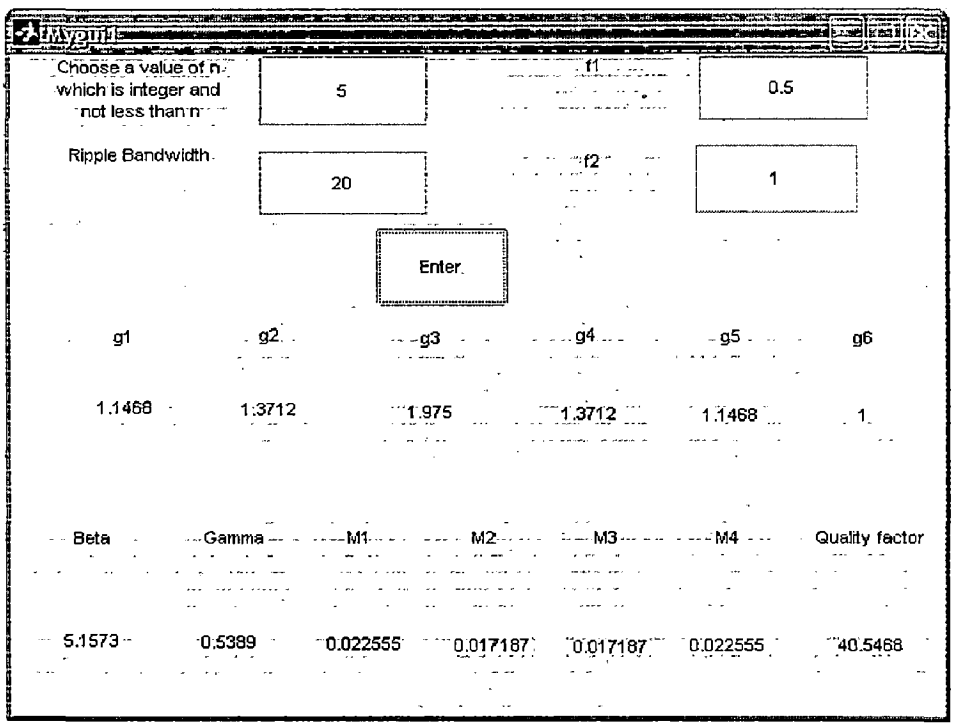


Fig. 4.2: GUI for Calculation of Filter Coefficients

MATLAB code for plotting the graph to calculate the resonator length

```
function mygui2()

% y = zeros()

theta=0:1:90;

theta1=theta*pi/180;

y=10*log10((theta1.*tan(theta1))./((tan(theta1)+theta1.*(1+(tan(theta1)).^2
))));

% end;

plot(theta,y);

axis([0   90   -22   -4]);

end
```

MATLAB code for calculating the resonator length

```
function varargout = Mygui3(varargin)

% MYGUI3 M-file for Mygui3.fig

%      MYGUI3, by itself, creates a new MYGUI3 or raises the existing

%      singleton*.

%

%      H = MYGUI3 returns the handle to a new MYGUI3 or the handle to

%      the existing singleton*.

%

%      MYGUI3('CALLBACK',hObject,eventData,handles,...) calls the local

%      function named CALLBACK in MYGUI3.M with the given input arguments.

%

%      MYGUI3('Property','Value',...) creates a new MYGUI3 or raises the

%      existing singleton*.  Starting from the left, property value pairs
are

%      applied to the GUI before Mygui3_OpeningFcn gets called.  An

%      unrecognized property name or invalid value makes property
application

%      stop.  All inputs are passed to Mygui3_OpeningFcn via varargin.
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Mygui3_OpeningFcn, ...
                   'gui_OutputFcn',  @Mygui3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Mygui3 is made visible.
function Mygui3_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for Mygui3
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
% --- Outputs from this function are returned to the command line.
function varargout = Mygui3_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function edit1_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit2_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

% --- Executes on button press in pushbutton1.

function pushbutton1_Callback(hObject, eventdata, handles)

deltaBLmin = get(handles.edit1,'string');

deltaBLmin = str2num(deltaBLmin);

deltaBLmax = get(handles.edit2,'string');

deltaBLmax = str2num(deltaBLmax);

f1 = get(handles.edit3,'string');

f1 = str2num(f1);

f2 = get(handles.edit4,'string');

f2 = str2num(f2);

thetamin1 = get(handles.edit5,'string');

thetamin1 = str2num(thetamin1);

thetamin2 = get(handles.edit6,'string');

thetamin2 = str2num(thetamin2);

%for calculating the resonator length

%deltaBLmin=20; %user has to provide

%deltaBLmax=50;  %user has to provide

f0=sqrt(f1*f2);

deltagamma=10*log10(deltaBLmax/deltaBLmin);

fprintf('%s%4.4f\n','deltagamma = ',deltagamma)

%thetamin1=12; %calculated from graph
```

```
%thetamin2=83; %calculated from graph

R1=thetamin1*(f0/f1);

R2=thetamin2*(f0/f2);

fprintf('%s%4.4f\n','Left bound of theta0=',R1)

fprintf('%s%4.4f\n','Right bound of theta0= ',R2)

set(handles.text4,'string',num2str(R1));

set(handles.text6,'string',num2str(R2));

set(handles.text12,'string',num2str(deltagamma));

function edit3_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit4_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit5_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit6_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit6_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end
```

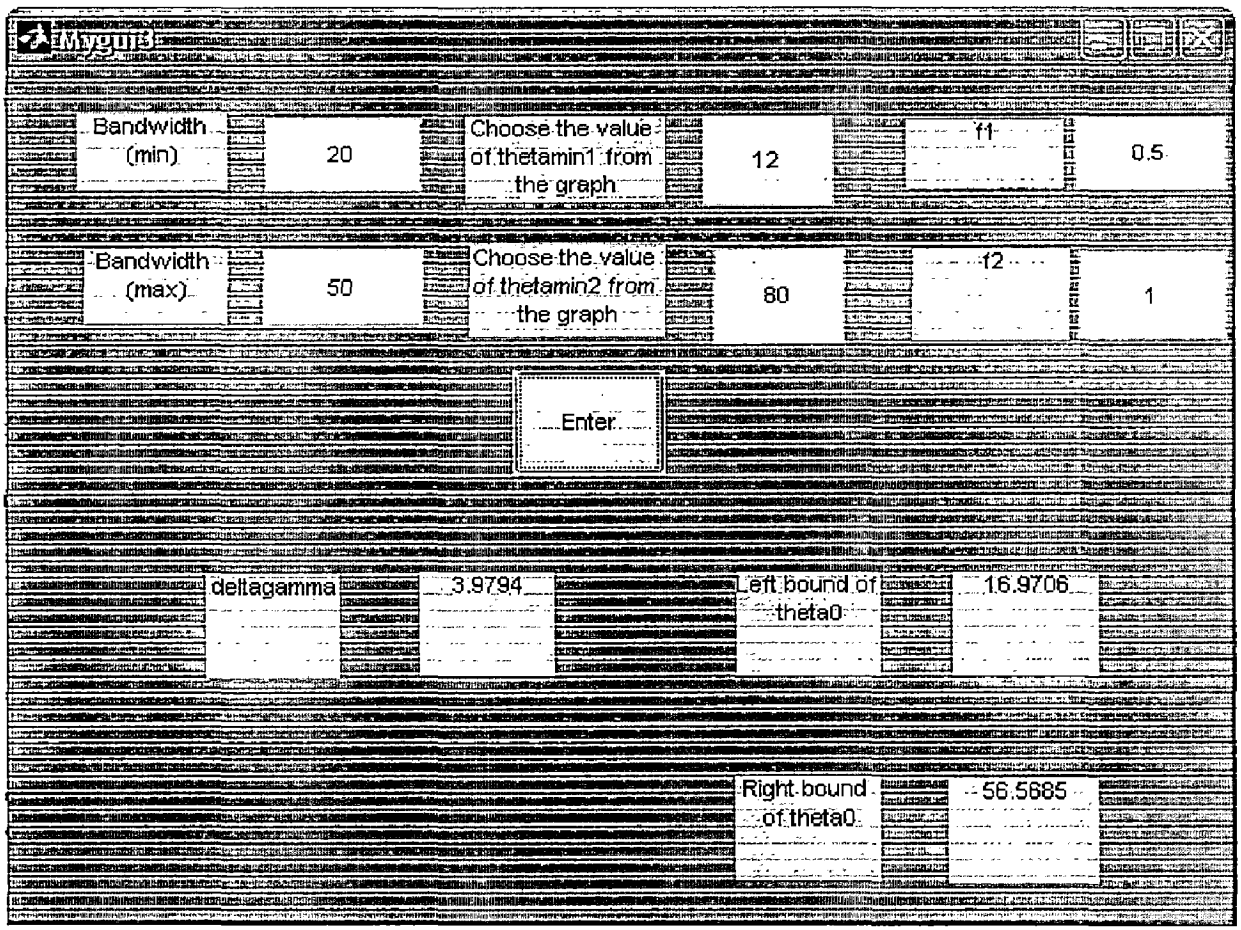


Fig. 4.3: GUI for Calculating Resonator's Electrical Length

MATLAB code for plotting the graph to calculate the tuning device capacitance

```
function Mygui4

% for calculating the tuning capacitance

f1=.5;f2=3;

Csmin=0.1;Csmax=1.2;theta0=36;

f0=sqrt(f1*f2);

f=f1:0.1:f2;

Cs0=f.*Csmin.*tan((theta0*pi.*f)./(f0*180))./(f0*tan(theta0*pi/180));

Cs1=f.*Csmax.*tan((theta0*pi.*f)./(f0*180))./(f0*tan(theta0*pi/180));

plot(Cs0,f);
```

```
axis([Csmin Csmax f1 f2]);
hold on;
plot(Cs1,f);
axis([Csmin Csmax f1 f2]);hold off;
end
```

MATLAB code for calculating the instantaneous bandwidth

```
function varargout = Mygui5(varargin)
% MYGUI5 M-file for Mygui5.fig
%      MYGUI5, by itself, creates a new MYGUI5 or raises the existing
%      singleton*.
%      MYGUI5('Property','Value',...) creates a new MYGUI5 or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Mygui5_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Mygui5_OpeningFcn via varargin.
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Mygui5_OpeningFcn, ...
                   'gui_OutputFcn',  @Mygui5_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Mygui5 is made visible.
function Mygui5_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for Mygui5
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes Mygui5 wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = Mygui5_OutputFcn(hObject, eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;
function edit1_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit2_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit3_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit4_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit5_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

% --- Executes on button press in pushbutton1.

function pushbutton1_Callback(hObject, eventdata, handles)

theta0 = get(handles.edit1,'string'); theta0 = str2num(theta0);

thetamin1 = get(handles.edit2,'string'); thetamin1 = str2num(thetamin1);

thetamax = get(handles.edit3,'string'); thetamax = str2num(thetamax);

deltaBLmin = get(handles.edit4,'string'); deltaBLmin = str2num(deltaBLmin);

deltaBLmax = get(handles.edit5,'string'); deltaBLmax = str2num(deltaBLmax);

%for calculating the instantaneous bandwidth deltaBL0

deltaBLn_theta0=(theta0.*tan(theta0))./((tan(theta0)+theta0.*(1+(tan(theta0
)).^2)));%theta is thetamin1

deltaBLn_theta=(thetamin1.*tan(thetamin1))./((tan(thetamin1)+thetamin1.*(1+
(tan(thetamin1)).^2)));

%theta is thetamax which is always 53 degrees

thetamax=53;
```

```
deltaBLn_thetamax=(thetamax.*tan(thetamax))./((tan(thetamax)+thetamax.*(1+(
tan(thetamax)).^2)));

A1=(deltaBLmin*deltaBLn_theta0)/deltaBLn_theta;

A2=(deltaBLmax*deltaBLn_theta0)/deltaBLn_thetamax;

%printf('%s%4.4f\n','Left bound of instantaneous bandwidth=',A1)

%printf('%s%4.4f\n','Right bound of instantaneous bandwidth=',A2)

set(handles.text13,'string',num2str(A1));

set(handles.text12,'string',num2str(A2));

function edit6_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.

function edit6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end
```

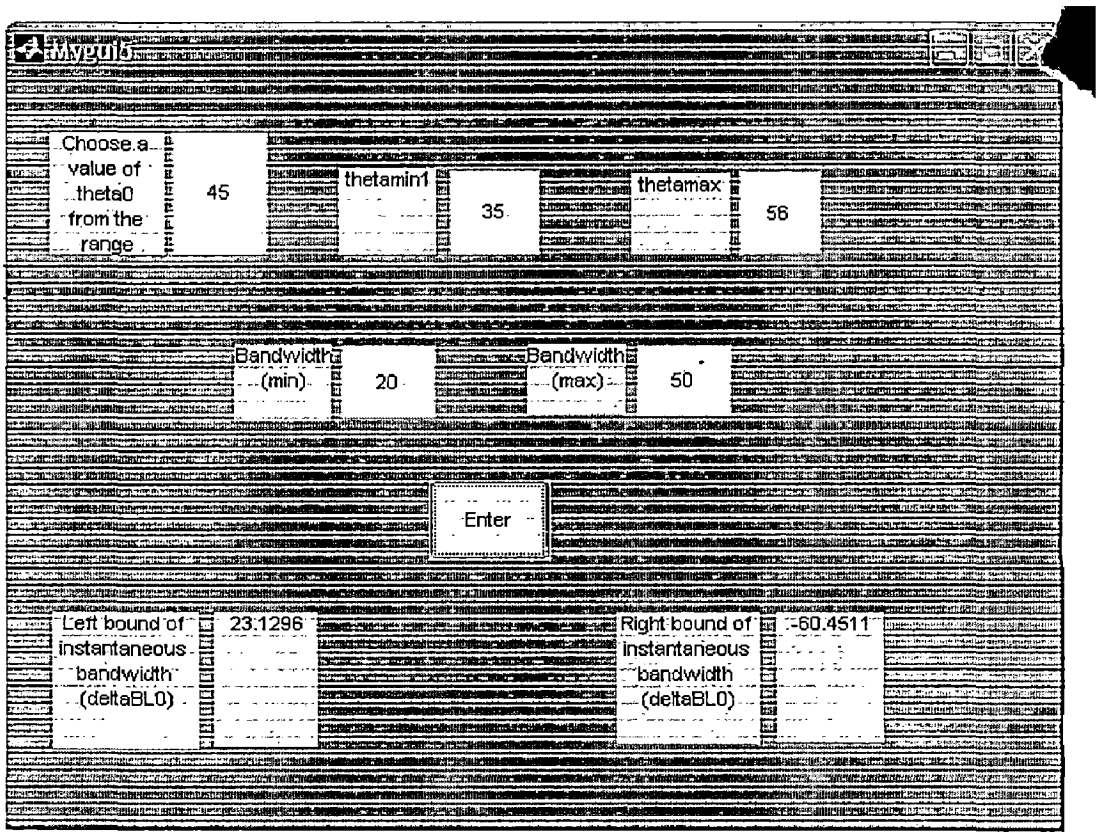


Fig. 4.4: GUI for Calculating the Filter's Instantaneous Bandwidth

Chapter 5
Conclusion and Future Scope

Conclusion

In this work step by step design procedure and implementation of Electronically Tunable combline filter with varactors is presented. The parasitic effect of the varactor has a profound impact on filter insertion loss. Increasing the length of the resonators will decrease the effect of the parasitic inductance of the varactors.

Future Scope

As we have seen in the design procedure bandwidth within the filter tuning range is not constant. We have also seen that coupling coefficient (K) is inversely proportional to the tuning frequency and external quality factor (Q_e) is directly proportional to the centre frequency .If we can able control these two parameters then we can achieve nearly constant bandwidth within the tuning range. Coupling coefficient (K) can be controlled by placing the coupling reducers within the resonators of the filter [27-28]. As we have seen the resonator quality factor (Q) is limited by the varactor losses. The varactor losses are compensated by coupling a negative resistor in the tuned circuit, the negative resistor is realized using an active device [29]. Tuning of the filter can also be done with Ferroelectric materials like BST (Barium Strontium Titanate),$Batio_3$ which have high tuning speed compared to varactors [30].

REFERENCES:

[1] Jeongpyo Kim and Jaehoon Choi, "Varactor Tuned Microstrip Bandpass filter with Wide Tuning Range," *Microwave and Optical Technology Letters*, Vol. 50, No. 10, pp. 2574-2577, Feb. 2008

[2] Friedrich K. Jondral, "Software Defined Radio-Basics and Evolution to Cognitive Radio," *EURASIP Journal on Wireless Communications and Networking*, Vol. 3, pp. 275-283, Apr. 2005

[3] M. Makimoto and S.Yamashita, *"Microwave Resonators and Filters for Wireless Communication: Theory, Design and Application"*. New York: Springer 2001

[4] A. Kozyrev, A. Ivanov, V. Keis, V. Osadchy, et. al., " Ferroelectric Films: Non-Linear Properties and Applications in Microwave Devices," *IEEE MTT-S Digest*, pp. 985-988, Dec. 1998

[5] H. Lee and S. Yun, "Microwave Planar Varactor Tuned Bandpass Filters: Historical Overview", *IEICE Trans. Electron.*, Vol. E89-C, No. 12, pp. 1806-1813, Dec. 2006

[6] D. M. Pozar, "*Microwave Engineering*", John Wiley, 2005

[7] J. G. Hong and M. J. Lancaster, *Microstrip Filters for RF/Microwave Applications*. New York: Wiley, 2001

[8] G. L. Matthaei, L. Young and E. M. T. Jones, *Microwave Filters, Impedance-Matching Networks and Coupling Structures*. New York: McGraw-Hill, 1980

[9] C. Bowick, J. Blyler and C. Ajluni, "*RF Circuit Design*," 2nd edition, Newnes Publications, 2008

[10] R. Ludwig and P. Bretchko, "*RF Circuit Design: Theory and Applications,*" New Jersy: Prentice Hall 2000

[11] Stephen A. Maas, "*The RF and Microwave Circuit Design Cookbook*," Artech House Inc., 1998

[12] S. Caspi and J. Aldeman, "Design of Combline and Interdigital Filters with Tapped Line Input," *IEEE Trans. Microwave Theory Tech.*, Vol. 36, pp. 759–763, Apr. 1988

[13] C. Hunter and J. D. Rhodes, "Electronically Tunable Microwave Bandpass Filters," *IEEE Trans. Microwave Theory Tech.*, Vol. 30, pp. 1354–1360, Sept. 1982

[14] Pierre Jarry and Jacques Beneat, "*Advanced Design Techniques and Realizations of Microwave and RF Filters,*" IEEE Press, John Wiley and Sons, 2008

[15] J. S. Wong, "Microstrip Tapped-Line Filter Design," *IEEE Trans. Microwave Theory Tech.*, Vol. 27, pp. 44–50, Jan. 1979

[16] K. Puglia, "A General Design Procedure for Band-Pass Filters Derived from Low-Pass Prototype Elements: Part I,'' *Microwave Journal.*, Vol. 43, No. 12, pp. 22–38, Dec. 2000

[17] K. Puglia, "A General Design Procedure for Band-Pass Filters Derived from Low-Pass Prototype Elements: Part II,'' *Microwave Journal*, Vol. 44, No. 1, pp. 114–136, Jan. 2001

[18] Babak Kazemi Esfeh and Alyani Ismail, "Narrow Band Elliptic Bandpass Filter using Dual-Mode Microstrip Square Loop Resonator for WiMax Application," *Journal of Modern Applied Science,* Vol. 3, No. 9, Sept. 2009

[19] J. Uher and W. J. R. Hoefer, "Tunable Microwave and Millimeter-Wave Bandpass Filters," *IEEE Trans. Micro. Theory Tech.*, Vol. 39, No. 4, pp. 643–653, Apr. 1991

[20] I. Bahl and Bhartia, "*Microwave Solid State Circuit Design*", John Wiley & Sons, 1988

[21] Daniel G. Swanson, Jr., "Narrow Band Microwave Filter Design," *IEEE Microwave Magazine*. pp. 105-114, Oct. 2007

[22] G. T. Penalva, G. L. Risueno and J. I. Alonso, "A Simple Method to Design Wideband Electronically Tunable Combline Filters", *IEEE Trans. Microwave Theory Tech.*, Vol. 50, No. 1, pp. 172-177, 2002

[23] John B. Ness, "A Unified Approach to the Design, Measurement, and Tuning of Coupled-Resonator Filters", *IEEE Trans. Microwave Theory Tech.*, Vol. 46, No. 4, pp. 343-351, April 1998

[24] D. G. Swanson, Jr., "Grounding Microstrip Lines with Via Holes," *IEEE Trans. Microwave Theory Tech.*, Vol. MTT-40, pp. 1719–1721, Aug. 1992

[25] S. Wen and L. Zhu., "Numerical Synthesis Design of Coupled Resonator Filters.," *Progress In Electromagnetics Research*, PIER 92, 333–346, 2009

[26] MATLAB (R2008a), The Mathworks Inc.

[27] Byung-Wook Kim, Sang-Won Yun., "Varactor Tuned Combline Bandpass Filter using Step-Impedance Microstrip Lines," *IEEE Trans. Microwave. Theory Tech.*, Vol. 52, No. 4, Apr. 2004

[28] M. Sanchez-Renedo, R. Gomez-Garcia, J. I. Alonso and C. Briso-Rodriguez, "Tunable Combline filter with Continuous Control of Center Frequency and Bandwidth," *IEEE Trans. Microwave. Theory Tech.*, Vol. 53, No. 1, pp. 191-199, Jan. 2005

[29] Ian Lockerbie and Surinder Kumar, "*A Broadband Tunable Combline Filter Using Active Devices.*" Communication systems Research Group, Department of Electrical Engineering, University of Saskatchewan, Canada

[30] F. Abbas and L. E. Davis, "Tunable Microwave Components Based on Dielectric Non-Linearity by Using HTS/Ferro-electric Thin Films," *IEEE Trans. on Applied Superconductivity*, Vol. 5, No. 4, Dec. 1995

resonators that are tuned by changing the biasing current (typically hundreds of milli amperes). Advantages of such filters are multi-octave tuning range, spurious free response, low insertion loss and high quality factor (Q-factor), while their disadvantages are size, power consumption, tuning speed and incompatibility in integrated Systems. Electronically tunable filters typically employ variable capacitors that are controlled by applying a bias voltage, thereby tuning the resonator. These filters provide octave tuning range, compact size, fast tuning and compatibility to integrate front ends. Several technologies are used to provide variable capacitors in tunable filters such as semiconductor diodes (gallium arsenide, silicon and silicon germanium), ferroelectric thin-films and RF microelectromechanical (MEMS) switches. Among these technologies, semiconductor diodes suffer from poor power handling, non-linear behavior and low Q-factor at microwave frequencies. Barium Strontium Titanate (BST) ferroelectric thin films provide high tunability at room temperature and have a relatively high power handling capability [4]. They are easily implemented in integrated planar structures and enable compact designs but suffer from poor linearity hence limiting the dynamic range of systems.

Recently, RF MEMS technology has provided means of creating highly linear,

Low-loss electromechanical switches that either provide an open circuit with a small

Capacitance (10-80 Fermi Farads) or a short circuit with a small resistance (0.6-1ohms). RF MEMS switches have several distinct advantages such as very low insertion loss (0.05-0.2 dB), very high linearity (IIP3 > 60 dBm), extremely low power consumption due to electrostatic actuation and very high isolation. They are very suitable for highly integrated systems and have been widely used in miniature switchable filters, phase shifters, etc. The disadvantages of

RF MEMS are low power handling capability (< 1-2W), medium switching speed (3-100μs), reliability and requirement for packaging.

1.4. Microwave Planar Varactor Tuned Bandpass Filters [5]

The varactor tuned bandpass filters can have various configurations such as combline, interdigital, coupled microstrip ring and hairpin and coupled line configurations. Even if there exist slight differences in the design, the standard bandpass filter design scheme is generally applied.