

75202

**AKIM TAŞIYICI ELEMAN KULLANARAK
SC AKTİF FİLTRE TASARIMI**

YÜKSEK LİSANS TEZİ

Müh. Hakan GÜRKAN

75202

Tezin Enstitüye Verildiği Tarih : 8 Haziran 1998

Tezin Savunulduğu Tarih : 30 Haziran 1998

Tez Danışmanı : Prof.Dr. Ali Nur GÖNÜLEREN

Diğer Jüri Üyeleri : Prof.Dr. Fuat ANDAY

Prof.Dr. Hakan KUNTMAN

17/07/1998
20/7/98
28/7/98

HAZİRAN 1998

ÖNSÖZ

Bu konuda çalışmamı sağlayan ve desteklerini esirgemeyen sayın hocam Prof. Dr. Ali Nur Gönülenen' e teşekkürü bir borç bilirim.

Haziran 1998

Hakan Gürkan



İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ.....	ii
ŞEKİL LİSTESİ.....	v
TABLO LİSTESİ.....	x
ÖZET	xi
SUMMARY.....	xii
BÖLÜM 1 GİRİŞ.....	1
BÖLÜM 2 AKTİF SC FİLTRELER.....	2
2.1 Analog İşaretlerin SC-Filtreler Yardımı İle İşlenmesi.....	2
2.2 İdeal SC-Filtrelerin Çalışma Prensibi	3
2.3 Örneklenmiş Verilere Ait Dalga Şekilleri	8
2.4 SC Devrelere Ait Transfer Fonksiyonları Arasındaki Bağlıntılar.....	13
2.5 s ve z Domeni Arasındaki Frekans Dönüşümleri	15
2.5.1 Bilineer Dönüşüm.....	15
2.6 İşlemsel Kuvvetlendirici Kullanılarak Tasarlanan İkinci Dereceden Aktif SC Filtreler	17
2.7 Anahtarlı-Kapasite Akım Taşıyıcı Eleman Kullanılarak Tasarlanan SC Filtre Devreleri.....	23
2.7.1 Anahtarlı-Kapasite Akım Taşıyıcı	23
2.7.2 SCCC Yapı Bloğu İle Oluşturulan İntegratör Devresi	25
2.7.3 SCCC Yapı Bloğu İle Oluşturulan Türev Alıcı Devresi.....	27
2.7.4 SCCC Yapı Bloğu İle Oluşturulan İkinci Dereceden SC Aktif Filtreler	29
BÖLÜM 3 Akım Taşıyıcı SC Filtreler	33
3.1 Akım Taşıyıcı	33
3.2 CCII-SC Filtrelerde Kullanılan Elemanların Ve z Domeni Eşdeğerlerinin Devre Yapıları	34
3.3 CCII-SC Aktif Filtrelerin Sentezi	39
3.4 Birinci Dereceden CCII-SC Aktif Filtreler.....	40
3.4.1 İntegratör Yapısı	40
3.4.2 Türev Alıcı.....	45
3.5 İkinci Dereceden CCII-SC Aktif Filtreler.....	47
3.5.1 İkinci Dereceden CCII-SC Aktif Filtrelerin Genel Yapısı.....	47

3.5.2 İkinci Dereceden Alçak Geçiren Filtre Yapısı	58
3.5.3 İkinci Dereceden Yüksek Geçiren Filtre Yapısı	61
3.5.4 İkinci Dereceden Band Geçiren Filtre Yapısı	64
3.5.5 İkinci Dereceden Notch Filtre Yapısı	67
3.6 Duyarlılık Analizi	69
3.6.1 Genlik Fonksiyonu Duyarlılığı	71
3.6.2 Katsayı Duyarlılığı.....	74
BÖLÜM 4 Akım Taşıyıcı SC Filtreler İle İlgili Örnek Tasarımlar	79
4.1 Bilineer Dönüşüm Metodu Kullanılarak Butterworth Filtre Tasarım Yöntemi	79
4.2 Akım Taşıyıcı SC Aktif Filtrelerle İlgili Örnek Tasarımlar	83
4.2.1 Alçak Geçiren Filtre Devresi	83
4.2.2 Yüksek Geçiren Filtre Devresi	85
4.2.3 Band Geçiren Filtre Devresi	88
4.2.4 Üçüncü Dereceden Alçak Geçiren Filtre Devresi.....	92
4.2.5 Dördüncü Dereceden Band Geçiren Filtre Devresi	95
BÖLÜM 5 Akım Taşıyıcı İçeren SC Aktif Filtrelerin Bilgisayar Programı İle Tasarımını	99
SONUÇLAR VE ÖNERİLER.....	107
KAYNAKLAR	109
EK-A.....	110
EK-B	115
EK-C	120
EK-D.....	126
EK-E	129
EK-F.....	134
ÖZGEÇMİŞ	233

ŞEKİL LİSTESİ

Sayfa No

Figure 1	Sampled-data filter system for analog input and smooth analog output .	xii
Figure 2a	CCII subnetwork	xv
Figure 2b	Associated subgraph	xv
Figure 3	Realization of 2th-order voltage transfer function	xvii
Şekil 2.1	Analog giriş ve düzgün analog çıkış için örneklenmiş veri filtre sistemi	2
Şekil 2.2a	Basit SC devreler	4
Şekil 2.2b	İki fazlı üstüste binmeyen saatler	4
Şekil 2.3a	Örneklenmiş gerilim dalga formu	5
Şekil 2.3b	Çift bileşenlerine ayrılmış hali	5
Şekil 2.3c	Tek bileşenlerine ayrılmış hali	5
Şekil 2.4	Örneklenmiş veri dalga şekilleri ve bunların sırasıyla çift ve tek bileşenlerine ayrılmış hali	10
Şekil 2.5	Sıfıra dönüşlü gerilim dalga şeklinin elde edilmesinde kullanılan anahtar devreler	11
Şekil 2.6	Örnekleme- tutma devreleri	12
Şekil 2.7	Dört kapılı z domeni eşdeğer devresi	13
Şekil 2.8	s ve z Düzlemine ait kararlılık bölgeleri	17
Şekil 2.9	Genel ikinci dereceden aktif SC devre yapısı	18
Şekil 2.10	SCCC yapı bloğunun iç yapısı	23
Şekil 2.11	SCCC yapı bloğunun şematik gösterimi	23
Şekil 2.12	SCCC aktif elemanı ile oluşturulan integratör devresi	25

Şekil 2.13	SCCC yapı bloğu ile oluşturulan türev alma devresi	27
Şekil 2.14	SCCC yapı bloğu ile oluşturulan ikinci derece genel aktif SC filtre devresi	29
Şekil 2.15	SC elemanı dönüşümleri	31
Şekil 3.1	Akım taşıyıcı modeli	33
Şekil 3.2	Akım taşıyıcı eşdeğer devresi	34
Şekil 3.3	Lineer iletkenlik	35
Şekil 3.4	Evirmeyen gecikmeli iletkenlik	35
Şekil 3.5	Anahtarsız kapasite	36
Şekil 3.6	Akım taşıyıcı z domeni eşdeğer devresi	37
Şekil 3.7	Akım taşıyıcılar ile oluşturulan gerilim toplama devresi	39
Şekil 3.8	Şekil 3.7'deki devrenin işaret akış diyagramı	40
Şekil 3.9a	Birinci dereceden CCII-SC aktif filtre devre topolojisi	41
Şekil 3.9b	Birinci dereceden CCII-SC aktif filtre devresinin z domeni eşdeğer devresi	42
Şekil 3.10	Saat ve gerilim dalga formu	42
Şekil 3.11a	Birinci dereceden türev alıcı CCII-SC aktif filtre devre topolojisi	47
Şekil 3.11b	Birinci dereceden türev alıcı CCII-SC aktif filtre devresinin z domeni eşdeğer devresi	47
Şekil 3.12a	Genel ikinci dereceden z domeni aktif filtre transfer fonksiyonlarının işaret akış diyagramı	48
Şekil 3.12b	Genel ikinci dereceden z domeni aktif filtre transfer fonksiyonlarının işaret akış diyagramının farklı bir versiyonu	49
Şekil 3.13a	İkinci dereceden genel CCII-SC aktif filtre z domeni eşdeğer devre yapısı	50
Şekil 3.13b	İkinci dereceden genel CCII-SC aktif filtre devre yapısı	51
Şekil 3.14	$D(z) = 1 + b_1 z^{-1} + b_2 z^{-2}$ için kararlı kutup yerleşim üçgeni	56

Şekil 3.15	CCII-SC devresi için gerçekleştirilebilir kutup yerleşim bölgesi.....	57
Şekil 3.16	İkinci derece alçak geçiren aktif filtre fonksiyonunun işaret akış diyagramı	59
Şekil 3.17	İkinci derece alçak geçiren aktif filtre yapısı	60
Şekil 3.18	İkinci derece yüksek geçiren aktif filtre fonksiyonunun işaret akış diyagramı	62
Şekil 3.19	İkinci derece yüksek geçiren aktif filtre yapısı	63
Şekil 3.20	İkinci derece band geçiren aktif filtre fonksiyonunun işaret akış diyagramı	65
Şekil 3.21	İkinci derece band geçiren aktif filtre yapısı	66
Şekil 3.22a	Genlik transfer fonksiyonunun A kapasitesine göre duyarlılığının frekansla değişimi	76
Şekil 3.22b	Genlik transfer fonksiyonunun B kapasitesine göre duyarlılığının frekansla değişimi	76
Şekil 3.22c	Genlik transfer fonksiyonunun C kapasitesine göre duyarlılığının frekansla değişimi	76
Şekil 3.22d	Genlik transfer fonksiyonunun D kapasitesine göre duyarlılığının frekansla değişimi	76
Şekil 3.22e	Genlik transfer fonksiyonunun E kapasitesine göre duyarlılığının frekansla değişimi	77
Şekil 3.22f	Genlik transfer fonksiyonunun F kapasitesine göre duyarlılığının frekansla değişimi	77
Şekil 3.22g	Genlik transfer fonksiyonunun G kapasitesine göre duyarlılığının frekansla değişimi	77
Şekil 4.1a	Alçak geçiren filtrenin örneklenmiş veri domenindeki frekans karakteristiği	79
Şekil 4.1b	Alçak geçiren filtrenin analog domenindeki frekans karakteristiği	79
Şekil 4.2a	Yüksek geçiren filtrenin örneklenmiş veri domenindeki frekans karakteristiği	80
Şekil 4.2b	Aynalama yöntemi ile elde edilen alçak geçiren eşdeğeri	80

Şekil 4.3a	Band geçiren filtrenin örneklenmiş veri domenindeki frekans karakteristiği	81
Şekil 4.3b	Band geçiren filtrenin analog domenindeki frekans karakteristiği	81
Şekil 4.4	İkinci derece alçak geçiren filtre devre topolojisi	85
Şekil 4.5	İkinci derece yüksek geçiren filtre devre topolojisi	88
Şekil 4.6	İkinci derece band geçiren filtre devre topolojisi.....	91
Şekil 4.7	$z=-1$ ' deki sıfırı atılarak elde edilen ikinci derece band geçiren aktif filtre devre topolojisi.....	92
Şekil 4.8	Üçüncü dereceden alçak geçiren filtre devresi.....	94
Şekil 4.9	Dördüncü dereceden band geçiren filtre devresi.....	97
Şekil 5.1	Program çalıştırıldığında karşılaşılabilecek ilk menü ekranı.....	101
Şekil 5.2	Parametre menüsü	102
Şekil 5.3	Program tarafından hesaplanan alçak geçiren filtre fonksiyonu ve bu fonksiyona ilişkin kapasite değerleri.....	103
Şekil 5.4	Şekil 5.3'de görülen transfer fonksiyonuna ilişkin frekans eğrisi.....	103
Şekil 5.5	Program tarafından hesaplanan yüksek geçiren filtre fonksiyonu ve bu fonksiyona ilişkin kapasite değerleri.....	104
Şekil 5.6	Şekil 5.5'de görülen transfer fonksiyonuna ilişkin frekans eğrisi.....	104
Şekil 5.7	Program tarafından hesaplanan band geçiren filtre fonksiyonu ve bu fonksiyona ilişkin kapasite değerleri.....	105
Şekil 5.8	Şekil 5.7'de görülen transfer fonksiyonuna ilişkin frekans eğrisi.....	105
Şekil A.1	Devreye ilişkin çift ve tek saat fazları.....	110
Şekil A.2	Frekans karakteristiği	111
Şekil A.3	Devrenin girişine uygulanan işaret.....	112
Şekil A.4	Devrenin çıkışında elde edilen işaretler	113
Şekil A.5	Örneklenmiş giriş ve çıkış karakteristikleri	114
Şekil B.1	Devreye ilişkin çift ve tek saat fazları.....	115

Şekil B.2	Frekans karakteristiği.....	116
Şekil B.3	Devrenin girişine uygulanan işaret.....	117
Şekil B.4	Devrenin çıkışında elde edilen işaretler.....	118
Şekil B.5	Örneklenmiş giriş ve çıkış karakteristikleri	119
Şekil C.1	Devreye ilişkin çift ve tek saat fazları.....	120
Şekil C.2	Frekans karakteristiği.....	121
Şekil C.3	Devrenin girişine uygulanan işaret.....	122
Şekil C.4	$(1 + z^{-1})$ terimine sahip olan devrenin çıkışında elde edilen işaretler....	123
Şekil C.5	$(1 + z^{-1})$ terimine sahip olmayan devrenin çıkışında elde edilen işaretler.....	124
Şekil C.6	Örneklenmiş giriş ve çıkış karakteristikleri	125
Şekil D.1	Devreye ilişkin çift ve tek saat fazları.....	126
Şekil D.2	Frekans karakteristiği.....	127
Şekil D.3	Örneklenmiş giriş ve çıkış karakteristikleri	128
Şekil E.1	Devreye ilişkin çift ve tek saat fazları.....	129
Şekil E.2	Frekans karakteristiği.....	130
Şekil E.3	Devrenin girişine uygulanan işaret.....	131
Şekil E.4	Devrenin çıkışında elde edilen işaretler	132
Şekil E.5	Örneklenmiş giriş ve çıkış karakteristikleri	133

TABLO LİSTESİ

	<u>Sayfa No</u>
Table 1 Basic SC building block and z-domain admittance	xvi
Tablo 2.1 Bilineer Dönüşüm İle Elde Edilen Filtre Tiplerine İlişkin Tasarım Denklemleri	21
Tablo 3.1 Basit SC Yapıları ve z Domeni Eşdeğerleri.....	38
Tablo 4.1 Alçak Geçiren Filtre Gerçeklemesi.....	84
Tablo 4.2 Yüksek Geçiren Filtre Gerçeklemesi	87
Tablo 4.3 Band Geçiren Filtre Gerçeklemesi	90
Tablo 4.4 Üçüncü Dereceden Alçak Geçiren Filtre Gerçeklemesi	93
Tablo 4.5 Dördüncü Dereceden Band Geçiren Filtre Gerçeklemesi	96

ÖZET

Bu tez çalışmasının amacı, bugüne kadar işlemsel kuvvetlendirici kullanılarak tasarlanan SC aktif filtre devrelerini, ikinci kuşak akım taşıyıcı eleman (CCII) ile tasarlamaktır.

Genel ikinci derece z domeni transfer fonksiyonunun öncelikle işaret akış diyagramı elde edilmiştir. Akım taşıyıcı toplama devresi ve bu devrenin işaret akış diyagramı baz alınarak, elde edilen işaret akış diyagramından akım taşıyıcı içeren SC aktif filtrenin devre topolojisine geçilmiştir. Bu geçiş, işaret akış diyagramının dal kazançlarının mutlak değerleri devredeki kapasite değerlerini; işaret akış diyagramının dal kazançlarının işaretleri de akım taşıyıcıların akım kazançlarını belirleyecek şekilde yapılmıştır. Elde edilen bu devre özelleştirilerek alçak geçiren, yüksek geçiren, band geçiren ve notch tipi filtrelerin devre topolojileri çıkartılmış ve devrede yer alan elemanların değerleri, ilgili fonksiyonun katsayıları cinsinden belirlenmiştir. Ayrıca bu devrelerin kararlılık koşulları ve duyarlılık fonksiyonları incelenmiştir.

Bu biçimde elde edilen CCII-SC aktif filtre tasarımını bilgisayar ortamına aktarmak için C dilinde bir bilgisayar programı hazırlanmıştır. Bu program dijital karakteristikleri girilen alçak, yüksek, band geçiren filtrelerin transfer fonksiyonunu hesaplamaktadır. Daha sonra bu fonksiyonun katsayılarına göre eleman değerlerini bulmaktadır. Bulduğu bu değerleri devrenin dinamik aralığını ve devrede yer alacak minimum kapasite değerini dikkate alarak normalize etmektedir. Ayrıca transfer fonksiyonun frekans karakteristiğini de çizmektedir.

Son olarak birkaç filtre tasarımı hazırlanmış ve son bölümde örnek olarak sunulmuştur. Hazırlanan bu örnekler spice simülasyon programı yardımıyla analiz edilmişlerdir. Analiz sırasında kullanılan devreler ideal akım taşıyıcı ve ideal anahtar içerir. Analiz sonucunda tasarlanan filtrelerin istenen frekans karakteristiklerini sağladıkları görülmüştür.

DESIGN OF SWITCHED CAPACITOR ACTIVE FILTER USING CURRENT CONVEYOR

The system given in figure 1 shows the analog signal being passed through a continuous anti-aliasing filter, an input sample-and-hold circuit (S/H)_i which samples the band-limited analog input at intervals of $1/f_{c1}$, the switched capacitor filter, an output sample-and-hold circuit (S/H)_o which resamples the output of the SC filter at intervals $1/f_{cN}$, and a final continuous reconstruction filter which serves to smooth the sharp transitions in the sampled-data waveform.

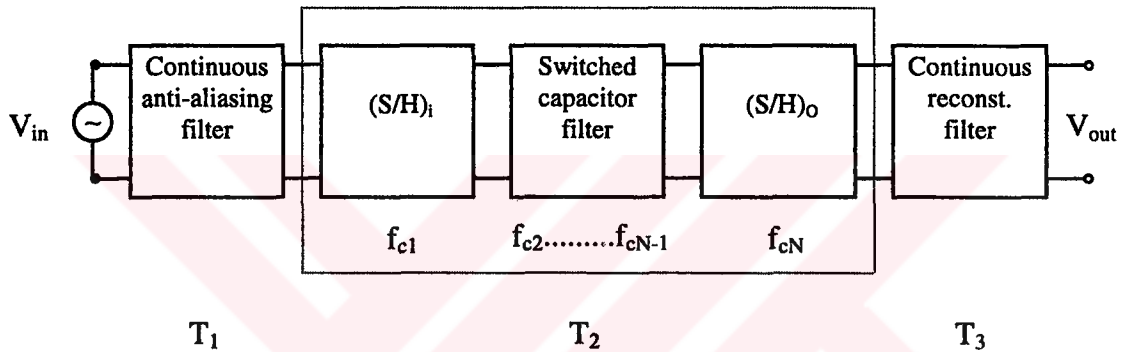


Figure 1 Sampled-data filter system for analog input and smooth analog output

The SC filter is shown to be controlled by clocks of multiple frequencies (f_{c2} through f_{cN-1}). To minimize the silicon area, it is often desirable to clock the low-pass sections at a high rate in order to lessen the burden on the continuous anti-aliasing filter, and to clock the high-pass sections at a lower rate, in order to reduce their total capacitance. Since the low-pass sections, sampled at f_{c2} , precede the high-pass sections which are sampled at $f_{c3} < f_{c2}$, the low-pass sections provide the needed anti-aliasing protection up to $f_{c2}/2$. It is noted that decreasing the sampling rate, typically requires no additional circuitry. However, increasing the sampling rate is a smoothing operation which typically requires additional low-pass filtering.

When the input or output is interfaced with digital or sampled-data circuitry, such as D/A or A/D converters, some of this hardware is no longer needed. For example, when the output is to be interfaced with a digital environment, the continuous reconstruction filter is no longer needed and the sampled-data circuit is typically incorporated with the digital circuitry. Although the needed for filtering is reduced, interfacing with digital or sampled-data circuits requires synchronization between the clocks that control the SC filter and those that control the external sampling operations. This is accomplished by passing synchronization pulses between the SC

network and external samplers. One reason for the synchronization is to ensure that the SC network output is sampled after all transients have settled and the output is truly held constant.

Consider the operation of an ideal SC network, comprised of ideal capacitors, ideal switches, and ideal voltage-controlled voltage sources etc. when excited by sampled-data voltage inputs.

Typically, the switches are controlled by a two-phases, nonoverlapping clock of frequency $f_c = 1/T$. Note that Φ^e is used to denote the even clock phase, which instantaneously closes the e switch on the even $2nT$ times. Similarly, Φ^o denotes the odd clock phase, which instantaneously closes the o switch on the odd $(2n+1)T$ times. The switches are assumed to have 50% duty cycle with equal (T second) on and off time periods. As the clock rate increase, the capacitor ratio, hence the silicon area, increases. Therefore, in practice the clock rate is typically chosen no higher than is required to achieve the desired degree of anti-aliasing protection with a second-order continuous filter of sufficiently high cutoff frequency to render its main passband variation acceptably small. The 50% duty cycle assumption is merely for simplification. The behavior of switched capacitor filters is strongly dependent on the clock period and typically is insensitive to the duty cycle. In fact, in practice, to ensure that the e and o switches are never turned on simultaneously, the clocks are made nonoverlapping. It is noted that turning both the Φ^e and Φ^o switches off simultaneously does not affect the behavior of the circuit; however, turning both switches on will cause improper circuit function.

The input or output of the SC network are sampled-data signals which change in value only at the switching instants kT . Thus, in their most general form, the voltage source and the internal circuit voltage are assumed to be sampled at times kT and held over a one-half clock period interval ($T/2$). With this assumption, we can apply z-transform techniques to the general analysis and synthesis of SC network. The z-transform, $z = e^{s\tau}$, where s is the complex analog frequency variable and $\tau = 2T$ is the clock period, then provides us with a convenient means for performing frequency-domain analysis.

Therefore the z-transform will exactly indicate the input-output relationship of an SC filter. To restore analog character to z-transform frequency response, this response must be modified multiplicative $(\sin \omega\tau/2)/(\omega\tau/2)$, where τ is the sampling period. For high sampling rates, where $\omega\tau \ll 1$, the passband of the frequency response is left virtually unaffected.

The switching action provides a time-varying nature to the SC network. That this is an output observed will depend upon when, and how often, the output is sampled.

The SC networks are separated two different topology as the switches open and closed. One topology corresponds to the even clock phase and a second topology to the odd clock phase.

One way to interpret the relationship between the even and the odd topologies is to consider them topologically decoupled, with the states of one determining the initial conditions for the other. This interpretation results in two distinct circuits coupled together via dependent sources which establish the aforementioned initial conditions. This formulation has been found to be particularly convenient for computer-aided analysis. Another interpretation is to combine the even and odd networks topologically into a single z-domain equivalent circuit. In general, an n -port bi-phase SC network will require a $2n$ -port.

Since SC networks can be most rigorously characterized in terms of charge-transfer operations, discrete-time voltages $V_i(kT)$ and discrete-time charge variations or transfers $\Delta q_i(kT)$ are used as port variables. At the switching times, kT , charges are instantaneously redistributed, with the principle of charge conservation maintained at every node in the network. It is this principle that allow us to write nodal charge equations similar to the way Kirchhoff's current law is used in continuous network. In general, due to the biphase switching operation, two distinct, but coupled, nodal charge equations are required to characterize the charge conservation condition at a particular node for all times instants, kT ; one equation for the even sampling instants and a second equation for the odd sampling instants are required. These equations are written, for some node p , as follows :

$$\Delta q_p^e(kT) = \sum_{i=1}^{Mep} q_{pi}^e(kT) - \sum_{i=1}^{Mep} q_{pi}^o[(k-1)T] ; \quad \text{for } k \text{ an even integer} \quad (1a)$$

$$\Delta q_p^o(kT) = \sum_{i=1}^{Mop} q_{pi}^o(kT) - \sum_{i=1}^{Mop} q_{pi}^e[(k-1)T] ; \quad \text{for } k \text{ an odd integer} \quad (1b)$$

or equivalently in the z -domain,

$$\Delta Q_p^e(z) = \sum_{i=1}^{Mep} Q_{pi}^e(z) - z^{-1/2} \sum_{i=1}^{Mep} Q_{pi}^o(z) \quad (2a)$$

$$\Delta Q_p^o(z) = \sum_{i=1}^{Mop} Q_{pi}^o(z) - z^{-1/2} \sum_{i=1}^{Mop} Q_{pi}^e(z) \quad (2b)$$

where q_{pi}^e , q_{pi}^o and Q_{pi}^e , Q_{pi}^o denote, respectively, the instantaneous charges stored on the i th capacitor connected to node p for the even and odd kT time instants and their z -transforms. Also Mep and Mop denote, respectively, the total number of capacitors connected to the node p during the even and odd clock phases.

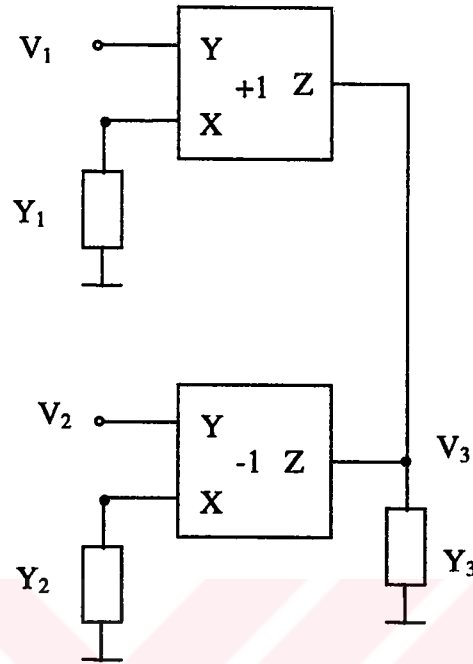
The initial step in the synthesis of an SC network is to obtain an appropriate z -domain transfer function. Since filters are typically specified by frequency-domain requirements, it is convenient to have a mathematical expression that allows us to transform rational s -domain transfer functions to rational z -domain transfer function. To be generally useful, such an expression should, in addition, possess two qualities :

1. Stable s -domain transfer functions map into stable z -domain transfer functions.
2. The imaginary $j\omega$ -axis of the s -plane map onto the unit circle of z -plane.

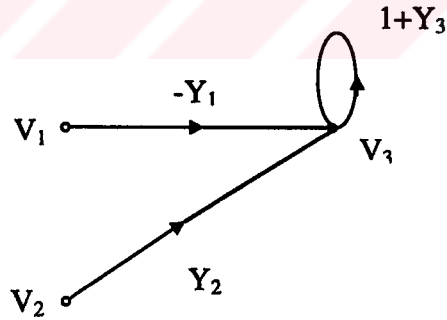
There are four transformation which have been used to synthesis SC network. These are back forward difference, forward difference, bilinear, lossless discrete integrator. The suitable transformation to convert s -domain transfer functions to z -domain transfer functions is bilinear transform.

Using z domain admittances of the basic SC building blocks, active SC filters which realize z -domain voltage transfer function, by using current conveyors and switched-capacitor, are projected. The synthesis method is based on drawing signal-

flow graph from the given transfer function and obtaining the circuit realization from the graph.



(a)

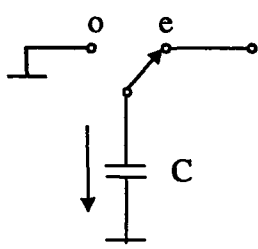
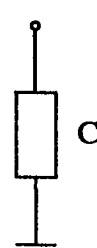
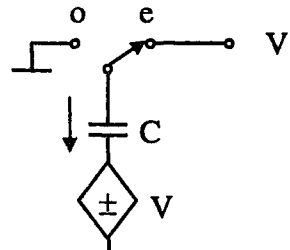
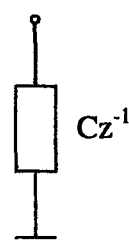
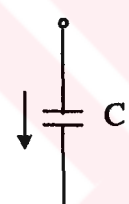
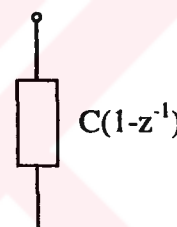


(b)

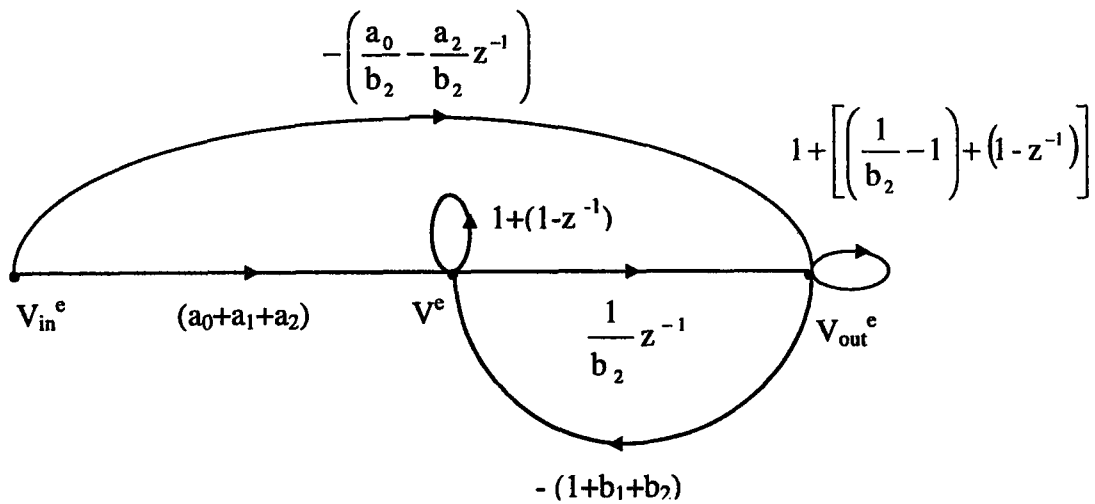
Figure 2 (a) CCII subnetwork
(b) Associated subgraph

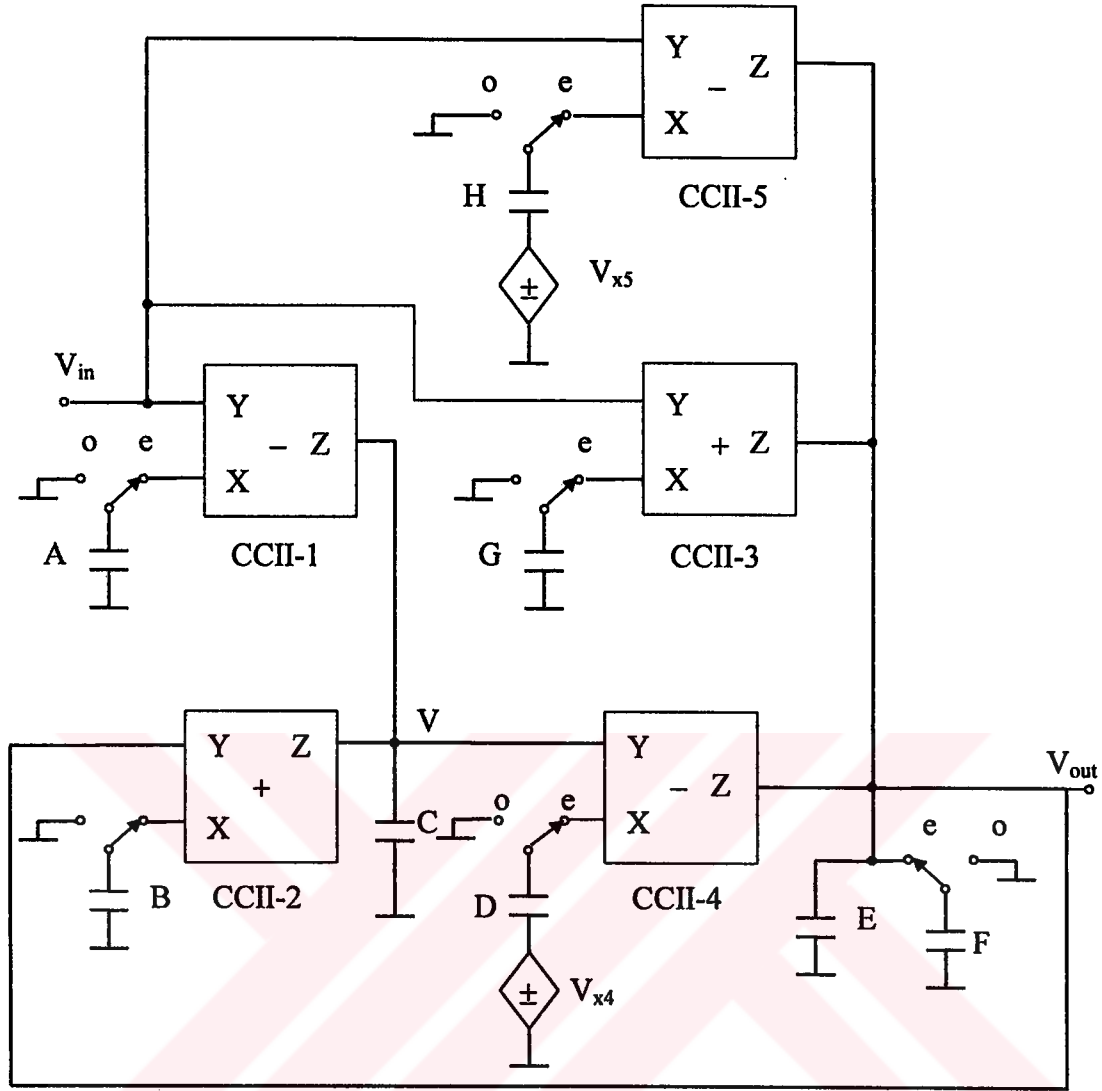
Consider the subnetwork shown in figure 2(a). Using the defining equation for the current conveyor, it can be shown that the subgraph in figure 2(b) corresponds to the subnetwork in figure 2(a). The z-domain equivalent admittances of the three basic SC elements can be obtained as shown in table 1.

Table 1 Basic SC building blocks and the z domain admittance

Circuit realization	Z domeni admittances
	
	
	

If a given z-domain transfer function can be represented by a suitable signal flow graph composed of subgraphs of the form of figure 2(b), the corresponding circuit realization can be easily found by the aid figure 2 and table 1.





(b)

Figure 3 Realization of 2th-order voltage transfer function

The realization of the 2th-order transfer function is considered. This transfer function

$$T(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (3)$$

can be represented by a signal-flow graph by using mason's gain formula as in figure 3(a). The circuit realization of this graph can be obtained as shown figure 3(b), with the aid of figure 2 and table 1.

BÖLÜM 1

GİRİŞ

Geçmiş yıllarda aktif SC devreler üzerine pek çok çalışma yapılmıştır. 1977 yılında J. T. Caves, M. A. Copeland, C. F. Rahim, S. D. Rosenbaum direnç eşdeğeri olarak anahtarlı kapasite kullanarak örneklenmiş analog filtreler tasarlamışlardır[1]. 1979 yılında C.F.Kurth ve G. S. Moschytz SC devrelerin analizinde yük denklemlerinin kullanılmasını önermişlerdir[2]. Aynı kişiler 1979' da iki kapılı SC devrelerin, dört kapılı z-domeni eşdeğer devresi kullanılarak analiz edilebileceğini ortaya koymuşlardır[3]. K. R. Laker yine aynı yıl SC devrelerin analiz ve sentezi için eşdeğer devreler geliştirmiştir[4]. 1980 yılında G. Szentirmai ve G. C. Temes genel ikinci derece örneklenmiş veri transfer fonksiyonlarını gerçekleyebilen SC yapı blokları ortaya koymuşlardır[5]. SC devrelerin z domeni eşdeğer devreleri özellikle aktif eleman içeriyorlarsa son derece karmaşıktır. 1984'de Z. X. Zhou herbir basit SC elemanını onun z domeni iletkenliği ile birebir değiştirerek aktif SC filtrelerin z domeni eşdeğer devrelerinin kolaylıkla elde edilmesini sağlayan bir yöntem geliştirmiştir[6]. 1987'de F. Anday z domeni gerilim transfer fonksiyonlarının işaret akış diyagramlarının oluşturulmasına ve bu diyagramdan, basit SC yapıları ve işlemsel kuvvetlendiricileri kullanarak devre gerçeklemesine dayanan bir sentez metodu geliştirmiştir[7]. 1993 yılında T.Ono yalancı akım taşıyıcı adını verdiği bir aktif eleman tasarlamış ve bu elemanla aktif SC filtre tasarımını gerçekleştirmiştir[8]. 1996 yılında da B.J.Maundy ve P.B.Aronhime anahtarlı kapasite ve işlemsel kuvvetlendirici kullanarak yeni bir akım taşıyıcı yapısı geliştirmişler ve bu yapıyı kullanarak integratör tasarımını gerçekleştirmişlerdir[9].

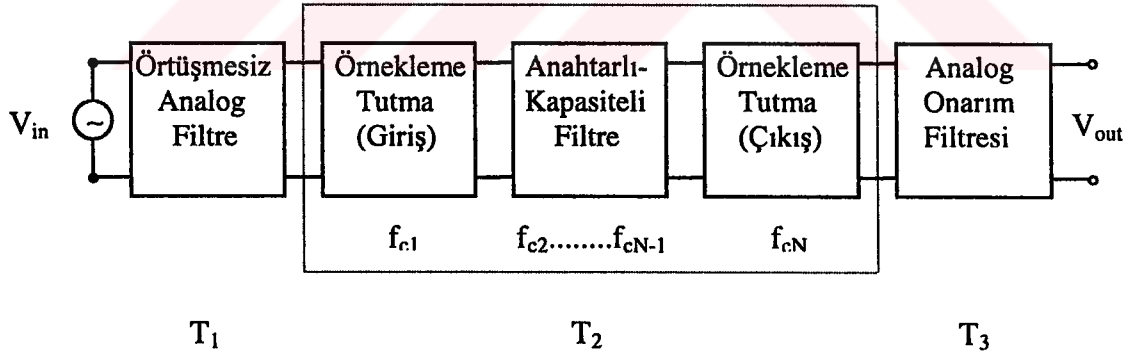
Bu çalışmada ise işaret akış diyagramlarından yararlanarak ikinci kuşak akım taşıyıcı kullanan SC aktif filtre devrelerinin nasıl tasarlanacağı incelenecektir.

BÖLÜM 2

AKTİF SC-FİLTRELER

2.1 Analog İşaretlerin SC-Filtreler Yardımı İle İşlenmesi

Sürekli zamandaki analog işaretlerin SC-filtreleri ile işlenebilmesi için ilk olarak işaretin örtüşmesiz band sınırlayıcı bir analog filtreden mutlak suretle geçirilmesi gerekir. İşaretin bozulmaması için örtüşme engellenmelidir. Bunun için örnekleme hızı, nyquist hızından bir başka deyişle işaretin band genişliğinin iki katından büyük olmalıdır. Aksi takdirde çıkışta elde edilecek sürekli işaret, giriş işaretine benzemez. Buna göre örnekleme hızının artırılmadığı durumlarda kesim frekansı, en yüksek örnekleme hızının yarısına eşit olan bir alçak geçiren filtre kullanılmalıdır. Burada yapılacak hata örtüşmeli duruma göre çok daha az olacaktır.



Şekil 2.1 Analog giriş ve düzgün analog çıkış için örneklenmiş veri filtre sistemi

Şekil 2.1’de görüldüğü gibi; örtüşmesiz analog filtreden geçirilerek bandı sınırlanan bir işaret, örnekleme frekansı f_{c1} olan bir örnekleme-tutma devresinin girişine uygulanarak örneklenir. Bu şekilde örneklenen işaret, anahtarlı-kapasiteli filtreden geçirilir. Hemen ardından bu kez çıkıştaki örnekleme-tutma devresinden

geçirilerek, SC filtre çıkışı $1/f_{cn}$ zaman aralıklarında örneklenir. Son olarak örneklenmiş işaret, onu tekrar analog hale getirecek analog onarım filtresinden bir başka deyişle bir alçak geçiren filtreden geçirilir. Böylece örneklenmiş işarettteki keskin geçişler yuvarlatılır ve çıkışta sürekli bir işaret elde edilir.

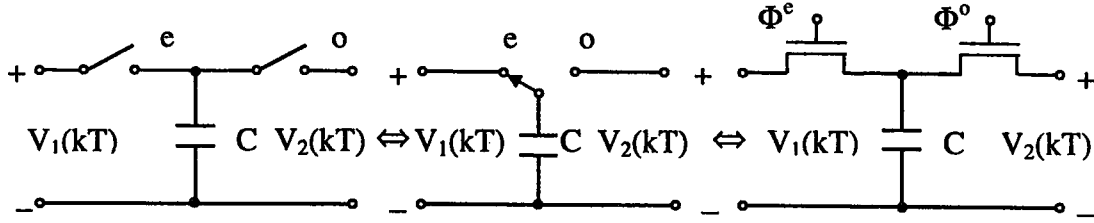
Yapının esasını oluşturan SC filtre f_{c2} 'den f_{cn-1} 'e kadar değişen frekanslar ile kontrol edilmektedir. Silikon yüzeyini ve örtüşmesiz filtreye gelen kısıtlamaları azaltmak için alçak geçiren bölgelere yüksek frekanslı saat darbeleri uygulanır. Bu arada, toplam kapasiteyi azaltmak için de yüksek geçiren bölgelere düşük frekanslı saat darbeleri uygulanır. Yapı gerçekleştirilirken alçak geçiren bölümler yüksek geçiren bölümlerden önce gelmek zorundadır. Alçak geçiren bölümler f_{c2} , yüksek geçiren bölümler f_{c3} ile örneklendiğinde $f_{c2} > f_{c3}$ olduğu için alçak geçiren bölümler $f_{c2}/2$ frekansına kadar örtüşmeden korunmuş olur. SC filtreye ait olan saat frekansları $f_{ci} = f_{MACL}/2^i$ bağıntısı ile bulunur. Buna göre ana saat frekansı f_{MACL} , frekans bölücü devrelerden geçirilerek f_{ci} frekansları elde edilir.

Şekil 2.1' de görülen devrenin girişi, çıkışı veya her ikisi birden sayısal bir sisteme bağlanacağı zaman sırasıyla band sınırlayıcı analog filtreye, analog onarım filtresine veya her ikisine birden gerek kalmaz. Örneğin devrenin çıkışı, sayısal bir sisteme bağlanacağı zaman çıkıştaki analog onarım filtresi çıkarılır ve örnekleme-tutma devresi direkt olarak sayısal sisteme bağlanır. Filtreleme için gerekenlerin azalmasına rağmen, SC filtrenin çıkışına sayısal veya örneklenmiş veri devreleri bağlanacağı zaman eş zamanlılığın mutlaka sağlanması gerekmektedir. Senkronizasyon, SC devresini kontrol eden saat darbeleri ile diğer devreleri kontrol eden saat darbeleri arasında yapılır. Senkronizasyonun amacı SC filtre çıkışının, tüm geçici durumlar sona erdiği ve kararlılığın sağlandığı anda örneklenmesini sağlamaktır.

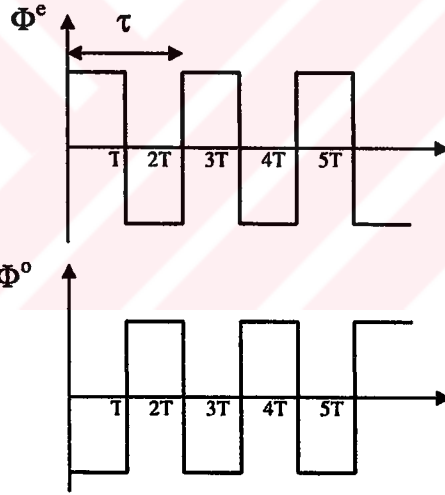
2.2 İdeal SC Filtrelerin Çalışma Prensipleri

SC devreleri ideal şartlarda düşünülürse, yapılarında bulundurdıkları kapasite, anahtar, gerilim kaynağı, işlemsel kuvvetlendirici, akım taşıyıcı gibi elemanların tümünün analog uyumluluğunun sağlanabilmesi için ideal olması gerekir. İşlemsel

kuvvetlendirici kullanılarak tasarlanan aktif SC filtrelerin yapılarında bulunan MOS işlemsel kuvvetlendiriciler pratikte son değerin %0.1' ine 2μ saniyede erişecek ve 60dB' den daha büyük doğru gerilim kazancı sağlayacak şekilde tasarlanmalıdırlar. Bu yüzden 250kHz' den daha düşük örnekleme hızları için, iyi bir yaklaşıklıkla, MOS işlemsel kuvvetlendiriciler a_0 kazançlı gerilim kontrollü gerilim kaynağı gibi davranırlar.



(a)



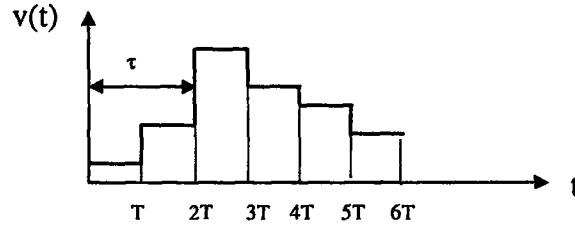
(b)

Şekil 2.2 (a) Basit SC devreler

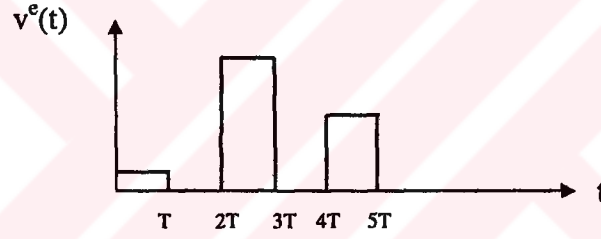
(b) İki fazlı üstüste binmeyen saatler

SC devrelerde kullanılan anahtar sembolleri Şekil 2.2(a)' da görülmektedir. Bu devrelerde kullanılan anahtarlar Şekil 2.2(b)' de görüldüğü gibi iki fazlı ve üst üste gelmeyen saat darbeleri tarafından kontrol edilir. Şekilde Φ^e ile gösterilen saat fazı,

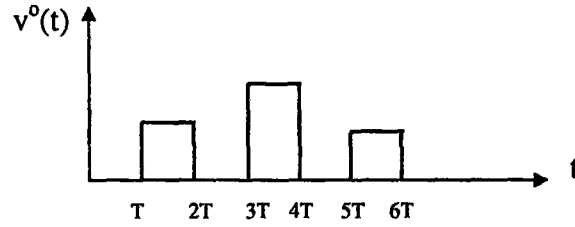
e anahtarını $(2nT)$ anlarında kapatırken, Φ^0 ile gösterilen saat fazı da o anahtarını $(2n+1)T$ anlarında kapatır. Şekilden de görülebileceği gibi anahtarların açık ve kapalı olduğu süreler birbirine eşit ve her sürenin ana periyodun %50' sine eşit olduğu farzedilmiştir. Genelde SC filtreler, 8kHz ile 26kHz arasındaki örnekleme frekanslarında çalışacak şekilde tasarlanırlar. Pratikte saat frekansı, örtüşmesiz analog filtrenin kesim frekansının iki katından daha büyük seçilemez.



(a)



(b)



(c)

Şekil 2.3 (a) Örneklenmiş gerilim dalga formu

(b) Çift bileşenlerine ayrılmış hali

(c) Tek bileşenlerine ayrılmış hali

Saat periyodunun %50'sinin anahtarlar tarafından tek ve çift zaman aralıklarında paylaşımı idealde yapılan bir kabuldür. Pratikte anahtarların aynı anda kapanmasından kaçınmak için anahtarlama zamanları örtüşmeyecek şekilde seçilmelidir. Bu nedenle anahtarlama süresi %50'den küçük alınabilir. Φ^e ve Φ^o anahtarlarının aynı anda kapanması devre davranışını etkilemez fakat aynı anda açılması devrenin istenmeyen şekillerde davranmasına neden olur. SC devrelerde, giriş ve çıkış işaretlerinin örneklenmiş değerlerinin sadece kT anahtarlama anlarında değiştiği düşünüldüğünden diğer iç düğümlerin ve kaynak gerilimlerinin de kT anlarında örneklendikleri ve yarım saat periyodu boyunca tutuldukları kabul edilir.

Buna göre SC devrelerin analizinde ve sentezinde kullanılacak en uygun yöntem z dönüşümü yöntemidir. z dönüşümü ile s dönüşümü arasında $z = e^{st}$ biçiminde lineer olmayan bir ilişki vardır. Frekans domeninde analizinde oldukça yarar sağlayan bu bağıntıda s, analog frekans bileşenini simgelerken, τ da saat periyodunu simgeler. Bu ifade görülen τ , 2T'ye eşittir.

Z dönüşümü SC filtrelerin giriş-çıkış ilişkisini tam olarak modeller. Bir SC filtrenin analog davranışı Şekil 2.3 (a)'da verilmiştir. Herhangi bir işaretin analog frekans cevabını, $(\sin(\omega\tau/2))/(\omega\tau/2)$ terimi ile çarparak bu cevabı z dönüşümü ile elde edilen frekans cevabına çevirmek gerekir. τ 'nın örnekleme periyodunu ifade ettiği bu terim, $z = e^{st}$ ilişkisinden kolaylıkla bulunabilir. Örnekleme hızı artırıldığında $\omega\tau < 1$ olacağından frekans cevabındaki geçirme bandı analog cevaptaki gibi aynen elde edilir.

Anahtarlama operasyonu SC devrelere zamanla değişme özelliği kazandırır. Bunun sonucu olarak da aktif SC devreler, anahtarların açık ve kapalı konumlarına göre iki farklı topolojiye ayrılır. Bunların biri çift saat fazına, ikincisi de tek saat fazına karşı düşer. Buna göre örneklenmiş dalga şekli, Şekil 2.3(b) ve (c)'de görüldüğü gibi çift ve tek bileşenlere ayrılabilir. Burada V^e ; çift saat fazı anındaki gerilimi V^o ; tek saat fazı anındaki gerilimi temsil eder.

Bu iki yapı çift ve tek devre yapısı olarak da adlandırılabilir. Bu yapılar arasındaki ilişki, herhangi birinin diğerinin başlangıç koşulunu belirleyecek şekilde tümleşik bir yapıyla ifade edilebilir. Bu yapılardan herhangi birisine ait ilk koşullar, bağımlı kaynaklar şeklinde gösterilir. Bu tip bir analiz metodu bilgisayar destekli

tasarımlar için oldukça elverişlidir.

Bu biçimde tarif edilen tek ve çift devreler tek bir z domenine eşdeğer devresine karşı düşürülebilir. Genel olarak n -kapılı, iki fazlı bir SC devresi, $2n$ -kapılı bir eşdeğer devreye karşı düşürülebilir. Bu eşdeğer devrede n -kapı tek saat fazı için diğer n -kapı da çift saat fazı içindir.

SC devrelerin çalışması yük transfer operasyonuna göre kesin olarak karakterize edilebileceğinden, kapı değişkenleri olarak ayrı zaman gerilimi $v_i(kT)$ ve ayrı zaman yük değişimi $\Delta q_i(kT)$ kullanılır. Anahtarlama anlarında yani kT anlarında, yüklerin korunumu ilkesine göre yükler, devrenin her bir düğümüne dağıtılır. Bu ilke doğrultusunda sürekli devrelerde kullanılan Kirchhoff akımlar yasasına benzer bir yolla düğüm yük denklemlerini yazabiliriz. Bir SC devresinde bir denklem, çift örnekleme anları için diğer bir denklem, tek örnekleme anları için olmak üzere iki ayrı düğüm yük denklemi verilmek zorundadır. Tüm kT anları için herhangi bir p düğümündeki düğüm yük denklemleri yazılacak olursa:

$$\Delta q_p^e(kT) = \sum_{i=1}^{Mep} q_{pi}^e(kT) - \sum_{i=1}^{Mep} q_{pi}^o[(k-1)T] \quad ; k \text{ çift tam sayı} \quad (2.1a)$$

$$\Delta q_p^o(kT) = \sum_{i=1}^{Mop} q_{pi}^o(kT) - \sum_{i=1}^{Mop} q_{pi}^e[(k-1)T] \quad ; k \text{ tek tam sayı} \quad (2.1b)$$

eşitlikleri elde edilir. Bu eşitliklerde q_{pi}^e ; p düğümüne bağlı i . kapasitenin çift, q_{pi}^o ise tek anlardaki yüklerini, Mep ve Mop ise sırasıyla p düğümüne bağlı çift ve tek anlardaki toplam kapasite miktarını simgeler. Burada τ örnekleme zamanının $2T$ 'ye eşit olduğunu unutulmamalıdır.

Herhangi bir $f(t)$ fonksiyonun laplace dönüşümü (2.2) eşitliğinde görülmektedir.

$$L\{f[(k-1)T]\} = F(s).e^{-sT} \quad (2.2)$$

Bu eşitlik dikkate alınarak (2.1) eşitliklerinin laplace dönüşümleri alınırsa (2.3) eşitlikleri elde edilir. Elde edilen bu eşitliklerde, $z = e^{sT}$ dönüşümü yapılırsa düğüm yük denklemlerinin (2.4) eşitliklerinde görülen z domenindeki ifadeleri elde edilir.

$$\Delta Q_p^e(s) = \sum_{i=1}^{Mep} Q_{pi}^e(s) - \sum_{i=1}^{Mep} Q_{pi}^o(s) e^{-s\tau} \quad (2.3a)$$

$$\Delta Q_p^o(s) = \sum_{i=1}^{Mop} Q_{pi}^o(s) - \sum_{i=1}^{Mop} Q_{pi}^e(s) e^{-s\tau} \quad (2.3b)$$

$$\Delta Q_p^e(z) = \sum_{i=1}^{Mep} Q_{pi}^e(z) - z^{-1/2} \sum_{i=1}^{Mep} Q_{pi}^o(z) \quad (2.4a)$$

$$\Delta Q_p^o(z) = \sum_{i=1}^{Mop} Q_{pi}^o(z) - z^{-1/2} \sum_{i=1}^{Mop} Q_{pi}^e(z) \quad (2.4b)$$

Tek kapasite içeren SC blokları için yazılan z domeni düğüm yük denklemlerine karşı düşen basit z domeni eşdeğer devreleri çizilebilir. Buradan hareketle karmaşık SC yapılar bloklara ayrılarak analiz edilebilirler.

2.3 Örneklenmiş Veri Dalga Şekilleri

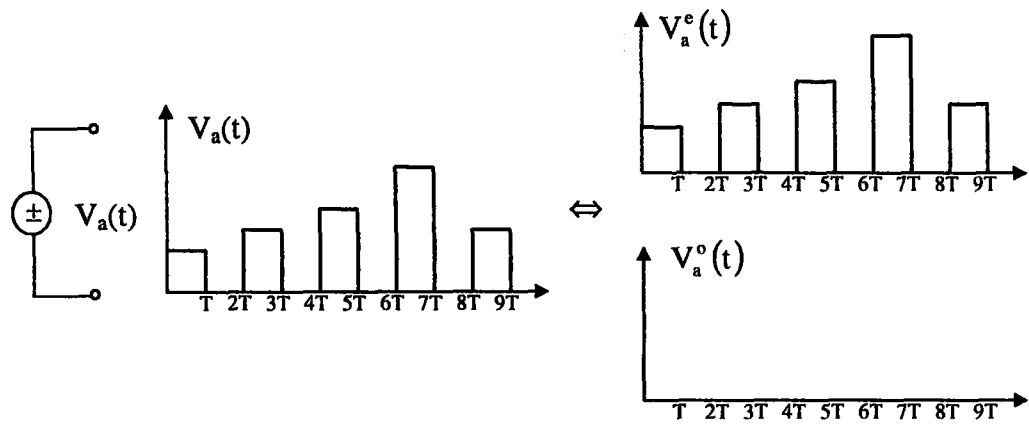
Özel durumlara karşı düşen gerilim dalga şekilleri ve bunların çift ve tek bileşenleri Şekil 2.4' de verilmiştir. Bunların z dönüşümleri alındığında Şekil 2.3 (a) ve (b)' deki sıfıra dönüşlü dalga şekilleri için matematiksel bağıntılar sırasıyla (2.5) ve (2.6)'daki gibi elde edilir.

$$V_a(z) = V_a^e(z) + V_a^o(z) \quad (2.5a)$$

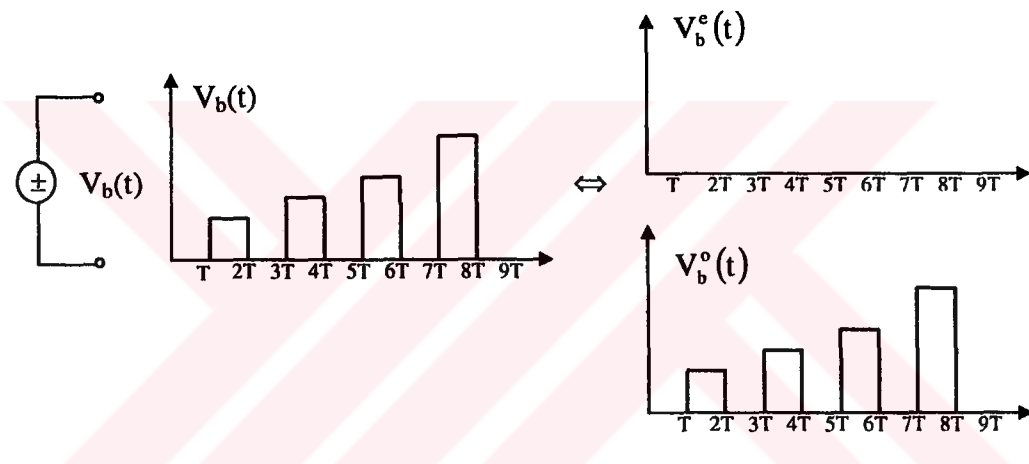
$$V_a^o(z) = 0 \quad (2.5b)$$

$$V_b(z) = V_b^e(z) + V_b^o(z) \quad (2.6a)$$

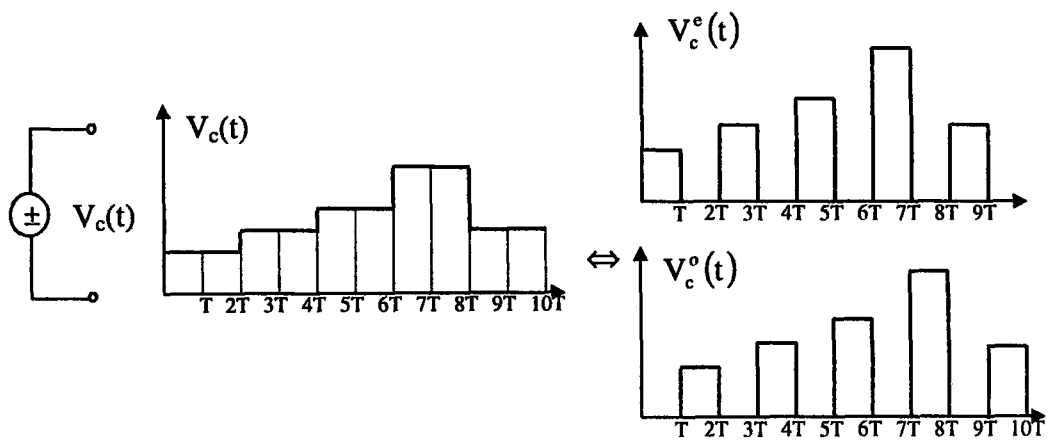
$$V_b^e(z) = 0 \quad (2.6b)$$



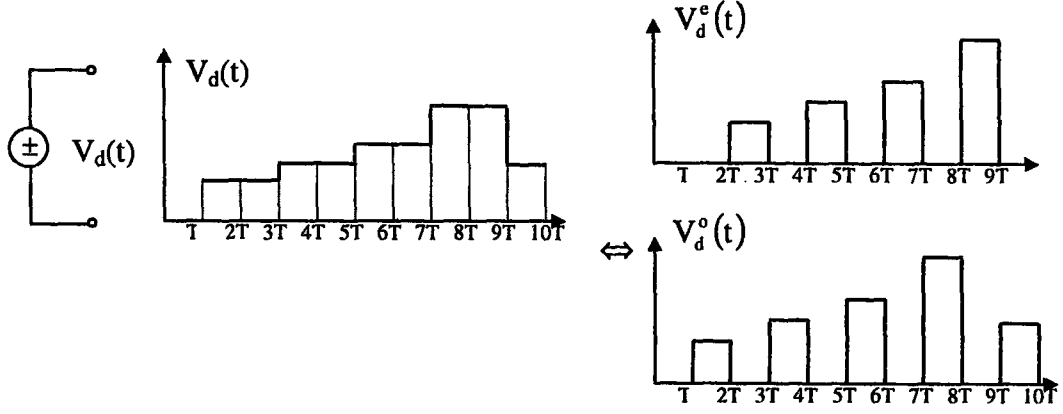
(a)



(b)



(c)



(d)

Şekil 2.4 Örneklenmiş veri dalga şekilleri ve bunların sırasıyla çift ve tek bileşenleri

Benzer yöntem Şekil 2.4(c)' deki tam saat periyotlu dalga şekillerine uygulanırsa

$$V_c(z) = V_c^e(z) + V_c^o(z) \quad (2.7a)$$

$$V_c^o(z) = z^{-1/2} V_c^e(z) \quad (2.7b)$$

(2.7) eşitlikleri elde edilir. Bu eşitlikten de görülebileceği gibi $V_c^o(kT)$, $V_c^e(kT)$ ' nin yarım periyotluk ötelenmiş değeridir. Son olarak Şekil 2.4 (d) görülen işaret için eşitlikler (2.8a) ve (2.8b)' de verilmiştir.

$$V_d(z) = V_d^e(z) + V_d^o(z) \quad (2.8a)$$

$$V_d^e(z) = z^{-1/2} V_d^o(z) \quad (2.8b)$$

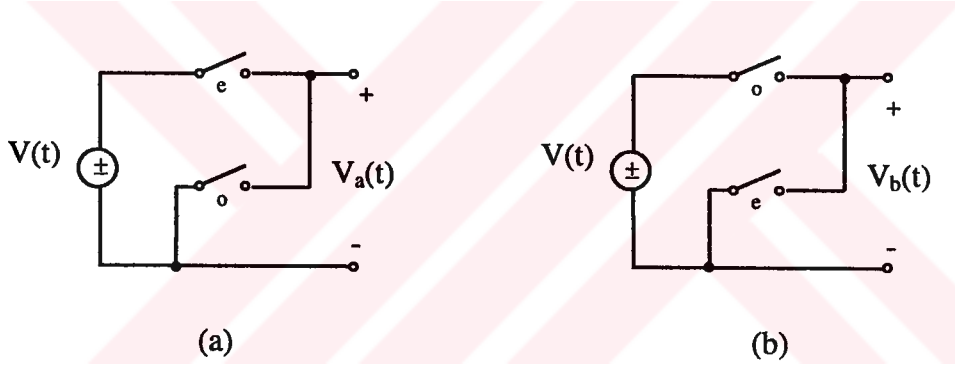
Eğer C değerinde bir kapasite, Şekil 2.4 (c)'de görülen $V_c(t)$ gerilim kaynağının uçları arasına bağlanırsa kapasite üzerindeki yükün sadece çift kT anlarında değiştiği, tek kT anlarında ise sabit kaldığı ve böylece yükün değişmediği gözlenir. Bu duruma ilişkin düğüm yük denklemleri tek ve çift saat fazları için yazılırsa (2.9)' da görülen

eşitlikler :

$$\Delta Q_c^e(z) = CV_c^e(z) - Cz^{-1/2}V_c^o(z) = C(1 - z^{-1})V_c^e(z) \quad (2.9a)$$

$$\Delta Q_c^o(z) = CV_c^o(z) - Cz^{-1/2}V_c^e(z) = 0 \quad (2.9b)$$

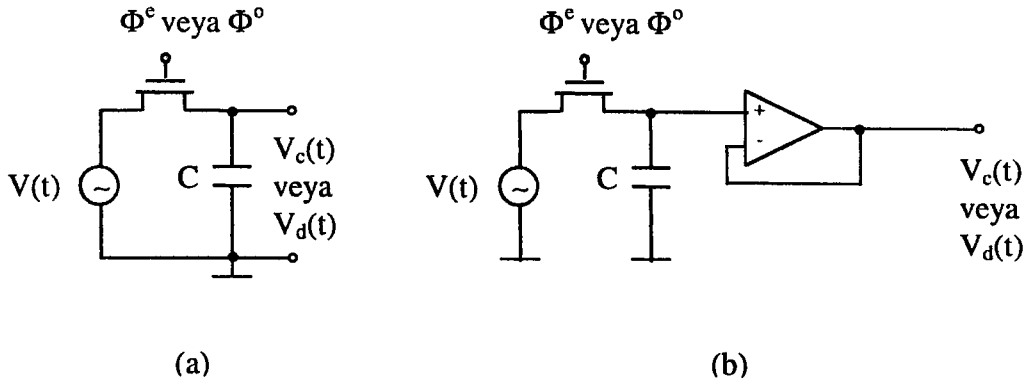
elde edilir. $\Delta Q_c^o(z) = 0$ koşulu, kapasite ile $V_c(t)$ gerilim kaynağı arasına bağlanan bir anahtarın tek saat fazı esnasında açılmasıyla elde edilir. Böylece tek saat fazı anında yük transferi olmaz. Bu açık devre koşulu da örnekleme ve tutma işlemi anlamına gelir. Benzer ilişki Şekil 2.4 (d)'deki işaret için de geçerlidir. Sıfıra dönüşlü $V_a(t)$ ve $V_b(t)$ dalga şekilleri, Şekil 2.3' de görülen $V(t)$ dalga şeklinin Şekil 2.5'de görülen basit anahtarlama devresine uygulanması ile elde edilir.



Şekil 2.5 Sıfıra dönüşlü gerilim dalga şeklinin elde edilmesinde kullanılan anahtar devreler

Şekil 2.4 (c) ve (d)'deki tam saat periyotlu ve Şekil 2.3 (a)'daki yarım saat periyotlu örnekleme-tutma dalga şekilleri, Şekil 2.6'da görülen örnekleme devresinin girişine sürekli işaret uygulanarak elde edilir. Devrede, normalde açık olan anahtar, örnekleme anlarında çok kısa bir süre kapatılarak veri işareti elde edilir. $V(t)$ geriliminin, anahtarın kapalı olduğu anlarda önemli ölçüde değişmemesi istendiğinden anahtarın açık ve kapalı tutulduğu süreler mümkün olduğunca kısa olmalıdır. Anahtar kapatıldığı zaman, C kapasitesi $V(t)$ geriliminin örnekleme anındaki değeri ile dolar. Anahtar açıldığı zaman C kapasitesi üzerindeki yük bir

sonraki örnekleme anına kadar sabit kalır. Şekil 2.6 (a)'daki basit örnekleme-tutma devresi, büyük değerli anahtarlı kapasite veya düşük değerli direnç gibi düşük empedanslı yüklere bağlanamaz. Bunun nedeni örnekleme anında, tutulan yükün düşük değerli empedans üzerinden boşalmasıdır. Bunu gidermek için kapasite ile yük arasına giriş direnci büyük, birim kazançlı bir kuvvetlendirici bağlanmalıdır.



Şekil 2.6 Örnekleme - tutma devreleri

Örnekleme-tutma işlemi, Şekil 2.6'daki devrenin girişine impuls uygulandığında çıkışından darbe elde edilmesiyle gerçekleşir. Anahtarın kapalı kaldığı periyot sırasında $V(t)$ geriliminin sabit kaldığı varsayılırsa devrenin impuls yanıtı:

$$h(t) = \frac{1}{\tau} \cdot u_1(t) - \frac{1}{\tau} \cdot u_1(t - \tau) \quad (2.10)$$

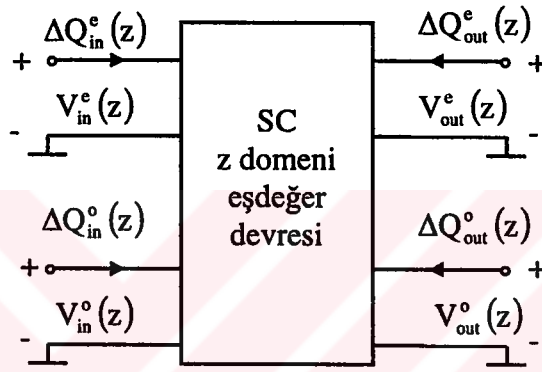
biçiminde hesaplanır. Bu denklemde $u_1(t)$ birim basamak fonksiyonunu, τ da iki örnek arasındaki zamanı ifade eder. Bu denkleme laplace dönüşümü uygulanırsa (2.11) eşitliği elde edilir. Bu eşitlikten elde edilen genlik ifadesi hem $\tau = T$ hem de $\tau = 2T'$ ye karşı düşen örnekleme işlemleri için tanımlı olup (2.12) eşitliğinde görülmektedir.

$$H(j\omega) = \left. \frac{1 - e^{-s\tau}}{s\tau} \right|_{s=j\omega} = \frac{\sin(\omega\tau/2)}{\omega\tau/2} \cdot e^{-j\omega\tau/2} \quad (2.11)$$

$$|H(j\omega)| = \frac{\sin(\omega\tau/2)}{\omega\tau/2} \quad (2.12)$$

2.4 SC Devrelere İlişkin Transfer Fonksiyonları Arasındaki Bağlılıklar

Aktif SC filtreler için z domeni gerilim transfer fonksiyonları, bu devrelerin tasarımında ve özelliklerinin belirlenmesinde önemli rol oynar. Biri giriş diğeri çıkış olmak üzere iki kapılı SC devreleri Şekil 2.7’de görülen 4-kapılı z domeni eşdeğer devre ile temsil edilebilir.



Şekil 2.7 Dört-kapılı z domeni eşdeğer devresi

Genel olarak dört-kapılı bir devrenin giriş çıkış ilişkisini (2.13)’de görülen 2×2 boyutunda bir matris tam olarak karakterize eder.

$$\begin{bmatrix} V_{out}^e(z) \\ V_{out}^o(z) \end{bmatrix} = \begin{bmatrix} H_1(z) & H_2(z) \\ H_3(z) & H_4(z) \end{bmatrix} \begin{bmatrix} V_{in}^e(z) \\ V_{in}^o(z) \end{bmatrix} \quad (2.13)$$

Toplamsallık ilkesine göre,

$$V_{in}(z) = V_{in}^e(z) + V_{in}^o(z) \quad (2.14a)$$

$$V_{out}(z) = V_{out}^e(z) + V_{out}^o(z) \quad (2.14b)$$

eşitlikleri yazılabilir. Çift kT anlarında $V_{in}^o(z)=0$ ve $V_{out}^o(z)=0$ olurken, tek kT anlarında da $V_{in}^e(z)=0$ ve $V_{out}^e(z)=0$ olur. (2.13) nolu denklem SC devrelerin zamanla değişebilirlik karakterini tamamen belirler. Bunun sonucunda çıkış, sadece çift kT anlarında örneklendiğinde (2.15a) eşitliği elde edilirken; tek kT anlarında örneklendiğinde de (2.15b) eşitliği elde edilir.

$$V_{out}^e(z) = H_1(z) \cdot V_{in}^e(z) + H_2(z) \cdot V_{in}^o(z) \quad (2.15a)$$

$$V_{out}^o(z) = H_3(z) \cdot V_{in}^e(z) + H_4(z) \cdot V_{in}^o(z) \quad (2.15b)$$

Eğer hem tek hem de çift anlarda örneklenirse çıkış (2.15c) eşitliğindeki gibi bulunur.

$$V_{out}(z) = V_{out}^e(z) + V_{out}^o(z) = [H_1(z) + H_3(z)] \cdot V_{in}^e(z) + [H_2(z) + H_4(z)] \cdot V_{in}^o(z) \quad (2.15c)$$

Genelde, $V_{out}^e(z) \neq z^{-1/2} \cdot V_{out}^o(z) \neq V_{out}(z)$ ' dir. Çift ve tek örnekler yarım saat periyotluk gecikmeyle birbirlerinden ayrılırlar. Bu yüzden tek ve çift bileşenler arasındaki ilişkiyi $V^o(z) = z^{-1/2} \cdot V^e(z)$ eşitliği gösterir. Bu da tam saat periyotlu örnekleme-tutma işlemine karşı gelir. Örnek olarak Şekil 2.4 (a)'daki sıfıra dönüşlü gerilim kaynağı Şekil 2.7'da görülen eşdeğer devreye uygulandığına $V_{in}^o(z)=0$ eşitliği de dikkate alınarak (2.15)' de görülen eşitlikler düzenlenirse (2.16) denklemleri :

$$V_{out}^e(z) = H_1(z) \cdot V_{in}^e(z) \quad (2.16a)$$

$$V_{out}^o(z) = H_3(z) \cdot V_{in}^e(z) \quad (2.16b)$$

elde edilir. O halde $V_{out}(kT)$ 'nin çift kT anlarında ya da tek kT anlarında örneklenmesine bağlı olarak gerilim transfer fonksiyonu ya $H_1(z)$ ya da $H_3(z)$ olacaktır. Fakat $V_{out}(kT)$ tüm kT anlarında örneklenecek olursa (2.17)'de görülen

eşitlik elde edilir. Genelde $H_1(z) \neq H_3(z)$ dir; fakat aynı bileşenleri içerirler.

$$V_{out}(z) = [H_1(z) + H_3(z)] \cdot V_{in}^e(z) \quad (2.17)$$

Bu nedenle $H_1(z)$ ve $H_3(z)$ birbirinden bağımsız olarak gerçekleştirilemezler.

2.5 S ve Z Domenleri Arasındaki Frekans Dönüşümleri

SC devre sentezinde yapılacak ilk iş uygun z domeni transfer fonksiyonlarının elde edilmesidir. Bunun için rasyonel s domeni transfer fonksiyonlarını rasyonel z domeni transfer fonksiyonlarına dönüştüren matematiksel bir yöntem ihtiyacı duyulur. Bu yöntemin sağlanması gereken koşullar :

1. Kararlı s domeni transfer fonksiyonlarını kararlı z domeni transfer fonksiyonlarına dönüştürmesi,
 2. s düzlemindeki sanal $j\omega$ eksenini z düzleminde birim daireye karşı düşürmesi,
- şeklinde özetlenebilir. Birinci özellik z domenine çevrilen transfer fonksiyonlarının, s domenindeki gibi kararlı olmasını sağlar. İkinci özellik ise transfer fonksiyonunu kararlı kılmakla beraber kazanç eğrisinin biçiminin s domenindeki gibi kalmasını sağlar.

SC devre sentezinde kullanılan belli başlı yöntemler geriye doğru fark dönüşümü, ileriye doğru fark dönüşümü, bilineer dönüşüm, kayıpsız ayrık integratör yöntemi ve impuls invaryans yöntemidir. Bu yöntemler incelendiğinde bilineer dönüşümün en uygun dönüşüm yöntemi olduğu görülür.

2.5.1 Bilineer Dönüşüm

Bilineer dönüşüm yamuk yöntemi ile integral veya türev almaya karşı düşer. (2.18)'de görülen denklem $t_0 = kT$ ile $t = (k+1) \cdot T$ arasında yamuk yöntemine göre integre edilirse :

$$\frac{dx}{dt} = -a \cdot x(t) \Rightarrow \int_{t_0}^t \frac{dx}{dt} \cdot dt = -a \cdot \int_{t_0}^t x(t) \cdot dt \quad (2.18)$$

(2.19)' da görülen eşitlik elde edilir. Bu eşitliğin her iki tarafının z dönüşümü alınırsa :

$$\int_{kT}^{(k+1)T} d(x(t)) = -a \cdot [x(kT) - x[(k+1)T]] \frac{T}{2} \quad (2.19a)$$

$$x[(k+1)T] - x[kT] = -a \cdot [x(kT) + x[(k+1)T]] \frac{T}{2} \quad (2.19b)$$

$$z \cdot X(z) - X(z) = -a \cdot \frac{T}{2} [z \cdot X(z) + X(z)] \quad (2.20a)$$

$$\frac{2}{T} \cdot \frac{1-z^{-1}}{1+z^{-1}} \cdot X(z) = -a \cdot X(z) \quad (2.20b)$$

(2.20a) ve (2.20b) eşitlikleri elde edilir. İfadenin $s \cdot X(s) = -a \cdot X(s)$ ile karşılaştırılması halinde bilineer dönüşüm bağıntısı (2.21)'de görüldüğü gibi elde edilir.

$$s = \frac{2}{T} \cdot \frac{1-z^{-1}}{1+z^{-1}} \quad (2.21)$$

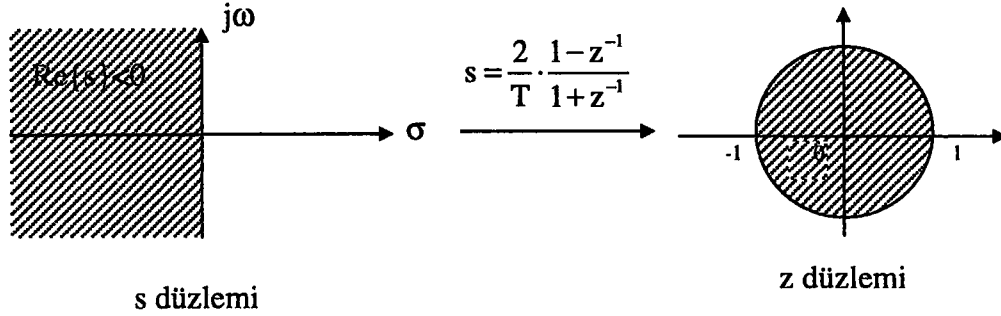
$$\text{Re}\{s\} < 0 \Rightarrow \text{Re}\left\{\frac{z-1}{z+1}\right\} < 0 \quad (2.22)$$

Kararlılık bölgesinin incelenmesine geçildiğinde (2.22) şartının belirlediği sınırları bulmak için $z = x + jy$ değeri (2.22) eşitliğinde yerine konursa (2.23) eşitsizliği elde edilir.

$$x^2 + y^2 < 1 \quad (2.23)$$

O halde kararlılık bölgesi birim çemberin iç bölgesi olmaktadır. (2.21) eşitliğinde de $s = j\omega$ değeri yerine konulursa $|z|$ değeri için;

$$|z| = \left| \frac{(2/T) + j\omega}{(2/T) - j\omega} \right| = 1 \quad (2.24)$$



Şekil 2.8 s ve z Düzlemine ait kararlılık bölgeleri

(2.24) eşitliği elde edilir. Her iki durumdan çıkarılan sonuç, daha önce belirtilen iki kararlılık koşulunu sağlıyor olmasıdır [10].

2.6 İşlemsel Kuvvetlendirici Kullanılarak Tasarlanan İkinci Dereceden Aktif SC Filtreler

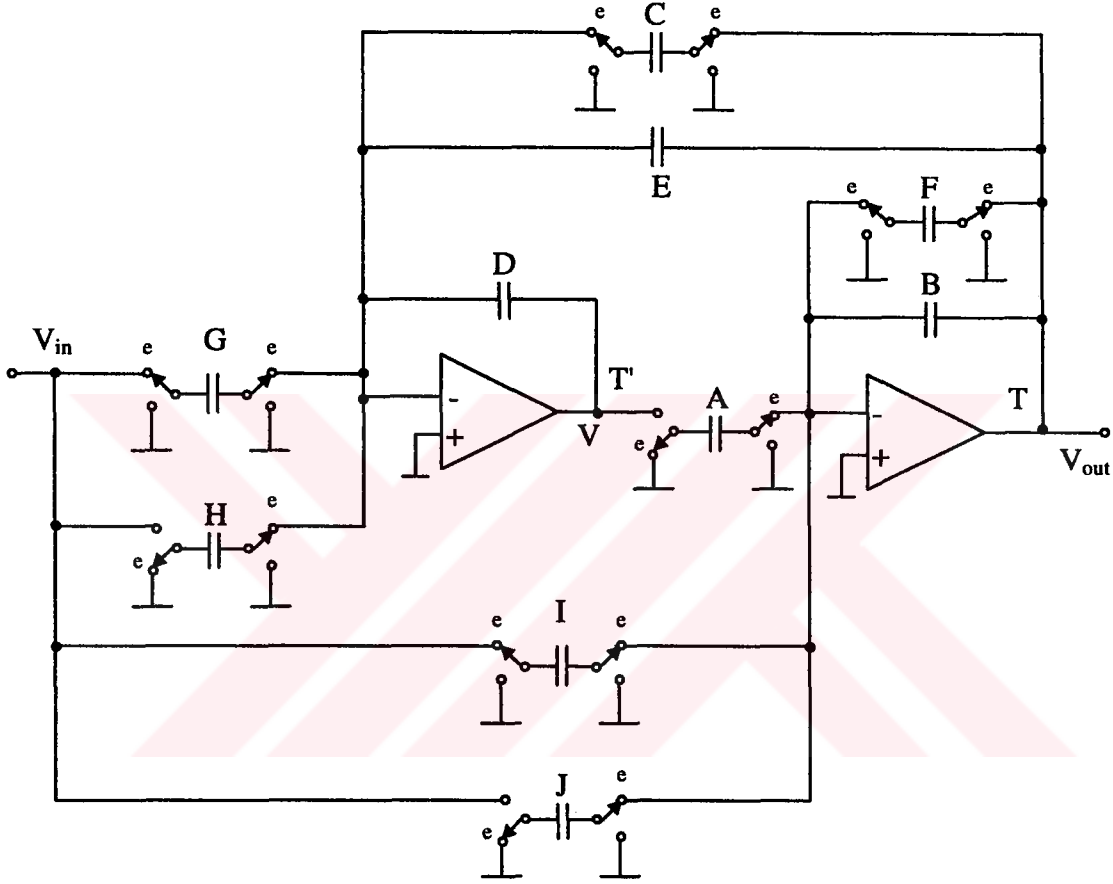
$$T(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (2.25)$$

Genel ikinci derece aktif-SC filtre devreleri (2.25)'de görülen z domenli transfer fonksiyonlarını gerçekleştirir. Şekil 2.9'da işlemsel kuvvetlendirici kullanılarak tasarlanan ve (2.25)'deki transfer fonksiyonunu gerçekleyen aktif-SC filtre devresi görülmektedir. Bu yapı, parazitik kapasitelerin etkisi yok edilmiş biçimde tasarlanmıştır.

Şekil 2.9'da verilen genel filtre yapısı iki alt bloktan meydana gelmektedir. Bu blokların birincisi evirici integratör olarak çalışırken, diğeri de evirmeyen integratör olarak çalışmaktadır. Devrede yer alan E ve F kapasitelerinin herhangi birinin atılması ile birbirinden farklı yapılar elde edilmektedir. Bu yapılar sırasıyla F-devresi (E=0), E-devresi (F=0) olarak adlandırılırlar.

Transfer fonksiyonun sıfırları, G, H, I ve J kapasitelerinin meydana getirdiği ileri

beslemeli yollar üzerinden gerçekleştirilir. Ayrıca bu kapasitelerin sadece ikisi kullanılarak ikinci derece transfer fonksiyonları gerçekleştirilebilir. Diğer bir ifade ile, ikinci dereceden transfer fonksiyonlarını gerçeklemek için yedi kapasiteye gereksinim vardır. Devrede yer alan Φ^0 ve Φ^e anahtar çifti; sembolik olarak iki anahtar yerine tek anahtar ile sadece çift saat (e) fazı için gösterilmiştir.



Şekil 2.9 Genel ikinci dereceden aktif SC devre yapısı

Bu yapıya ilişkin transfer fonksiyonunun elde edilmesi için bazı koşulların önceden belirlenmesi gerekir. İlk olarak analizde kolaylık sağlanması açısından SC devrelerin, asıl periyodun %50'sini paylaşan ve üstüste gelmeyen Φ^0 ve Φ^e saat fazlarını kullandıkları kabul edilmektedir. Bunun yanı sıra, giriş işaretinin örneklenen değerinin de tam saat periyodu boyunca tutulduğu kabul edilmektedir. Bu tespitlerin ışığında (2.26) eşitliği yazılabilir. İşlemsel kuvvetlendiricinin çıkışının da tam saat periyodu boyunca tutulduğu ve çift saat fazı anlarında değiştiği dikkate alınırsa (2.27)

eşitliği elde edilir.

$$V_{in}^o(z) = z^{-1/2} V_{in}^e(z) \quad (2.26)$$

$$V_{out}^o(z) = z^{-1/2} V_{out}^e(z) \quad (2.27)$$

Devreye ilişkin transfer fonksiyonunu bulmak için işlemsel kuvvetlendiricilerin eviren giriş düğümlerine ait düğüm-yük denklemleri kullanılır. Bu denklemlerde sadece çift saat fazı gözönüne alınmaktadır. O halde birinci ve ikinci işlemsel kuvvetlendiriciye ilişkin düğüm-yük denklemleri:

$$GV_{in}^e(z) + DV^e(z) + EV_{out}^e(z) + CV_{out}^e(z) - z^{-1/2} (HV_{in}^o(z) + DV^o(z) + EV^o(z)) \quad (2.28)$$

$$IV_{in}^e(z) + BV^e(z) + FV^e(z) - z^{-1/2} (JV_{in}^o(z) + AV^o(z) + BV^o(z)) \quad (2.29)$$

şeklinde elde edilir. (2.26) ve (2.27) eşitlikleri yukarıda görülen denklemlerde yerine konur ve gerekli düzenlemeler yapılırsa devreye ait T ve T' transfer fonksiyonları (2.30) ve (2.31) eşitliklerinde görüldüğü gibi elde edilir.

$$T(z) = \frac{V_{out}^e}{V_{in}^e} = - \frac{DI + (AG - DI - DJ) \cdot z^{-1} + (DJ - AH) \cdot z^{-2}}{D \cdot (F + B) + (AC + AE - DF - 2BD) \cdot z^{-1} + (DB - AE) \cdot z^{-2}} \quad (2.30)$$

$$T'(z) = \frac{V^e}{V_{in}^e} \quad (2.31)$$

$$= \frac{(IC + IE - GF - GB) + (FH + BH + BG - JC - JE - IE) \cdot z^{-1} + (EJ - BH) \cdot z^{-2}}{D \cdot (F + B) + (AC + AE - DF - 2BD) \cdot z^{-1} + (DB - AE) \cdot z^{-2}}$$

A kapasitesi T ve T' fonksiyonlarına ait kazanç katsayılarını birbirinden bağımsız olarak belirlemek için kullanılmaktadır. Böylece dinamik çalışma aralığı maksimum seviyeye çıkartılabilir. B ve D kapasiteleri de işlemsel kuvvetlendiricilerin toplama noktasındaki admitans seviyelerini kontrol etmektedir.

Bunun sonucu olarak (C, D, E, G, H) ve (A, B, F, I, J) olmak üzere iki kapasite grubuna T ve T' fonksiyonlarını deęiřtirmeden, birbirinden bağımsız olarak normalizasyon işlemi uygulanabilir. Devrenin sentezini kolaylařtırmak amacıyla A, B, D kapasiteleri bire eřit alınırsa (2.32) eřitliklerinde görülen basitleřmiř T ve T' fonksiyonları elde edilir.

$$T(z) = \frac{V_{out}^e}{V_{in}^e} = - \frac{I + (G - I - J) \cdot z^{-1} + (J - H) \cdot z^{-2}}{(F + 1) + (C + E - F - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}} \quad (2.32a)$$

$$T'(z) = \frac{V^e}{V_{in}^e} \quad (2.32b)$$

$$= \frac{(IC + IE - GF - G) + (FH + H + G - JC - JE - IE) \cdot z^{-1} + (EJ - H) \cdot z^{-2}}{(F + 1) + (C + E - F - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}}$$

Yukarıda görülen transfer fonksiyonları incelendiğinde, kutupların C, F, E kapasiteleri, sıfırlarının ise I, J, G, H kapasiteleri tarafından belirlendięi görülür. Pay polinomuna ait tüm katsayılar bağımsız olarak belirlenebilmektedir. Bu da keyfi sıfırların saęlanması imkanını tanımaktadır.

İkinci dereceden aktif SC filtreye ait transfer fonksiyonlarında $F = 0$ alınırsa elde edilen devreye E devresi, $E = 0$ alınırsa elde edilen devreye de F devresi ismi verilir. Bunlara karřı düşen yeni T ve T' fonksiyonları (2.33) eřitliklerinde görölmektedir.

$$T_E(z) = - \frac{I + (G - I - J) \cdot z^{-1} + (J - H) \cdot z^{-2}}{1 + (C + E - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}} \quad (2.33a)$$

$$T'_E(z) = \frac{(IC + IE - G) + (H + G - JC - JE - IE) \cdot z^{-1} + (EJ - H) \cdot z^{-2}}{1 + (C + E - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}} \quad (2.33b)$$

$$T_F(z) = - \frac{I + (G - I - J) \cdot z^{-1} + (J - H) \cdot z^{-2}}{(F + 1) + (C - F - 2) \cdot z^{-1} + z^{-2}} \quad (2.33c)$$

$$T'_F(z) = -\frac{(-IC + GF + G) + (-FH - H - G + JC) \cdot z^{-1} + H \cdot z^{-2}}{(F+1) + (C - F - 2) \cdot z^{-1} + z^{-2}} \quad (2.33d)$$

Yukarıda görülen transfer fonksiyonları incelendiğinde T_E ve T_F fonksiyonlarının paylarının aynı olduğu, buna karşın T'_E ve T'_F fonksiyonlarının paylarının farklı olduğu görülür. Elde edilmek istenen çıkışın V_{out} olduğu bir tasarımda, V çıkışı maksimum dinamik aralığı verecek şekilde devrede normalizasyon yapılmadığı zaman iki devre birbirinden oldukça farklı davranmaktadır. Aynı durum istenen çıkışın V olduğu durumda da geçerlidir. Bu da kapasitelerin normalize edilmiş değerlerini ve toplam kapasite değerini değiştirmektedir. Ayrıca bu dört devre arasında önemli ölçüde duyarlılık farklılıkları mevcuttur. E devresine ait transfer fonksiyonu kararlılık açısından incelendiğinde z düzleminde yer alan tüm kararlı kutupların gerçekleştirilebileceği görülmüştür. Ancak bu F devresi için geçerli değildir. F devresine ilişkin transfer fonksiyonu incelendiğinde paydada yer alan z^{-2} teriminin katsayısının (b_2) hiçbir zaman negatif olamayacağı görülür. Bu nedenle F devresi, bu katsayının negatif olduğu bölgede yer alan kutupları bir başka deyişle ters işaretli reel kutupları gerçekleyemez. Kararlı F devreleri kompleks veya aynı işaretli reel kutuplara sahip olabilirler.

Tablo 2.1 Bilineer dönüşüm ile elde edilen filtre tiplerine ilişkin tasarım denklemleri

Filtre Türü	Pay Polinomu	Tasarım Denklemleri	Basit Çözüm
LP	$K(1 + z^{-1})^2$	$I = K , J - H = K $ $G - I - J = 2 \cdot K $	$I = J = K , H = 0$ $G = 4 \cdot K $
HP	$K(1 - z^{-1})^2$	$I = K , J - H = K $ $G - I - J = -2 \cdot K $	$I = J = K ,$ $G = H = 0$
BP	$K(1 + z^{-1})(1 - z^{-1})$	$I = K , J - H = - K $ $G - I - J = 0$	$I = K , J = 0$ $G = H = K $
HPN ve LPN	$K(1 + \varepsilon z^{-1} + z^{-2})$	$I = K , J - H = K $ $G - I - J = \varepsilon \cdot K $	$I = J = K , H = 0$ $G = K \cdot (2 + \varepsilon)$
AP, $\beta > 0$	$K(\beta + \alpha z^{-1} + z^{-2})$	$I = K , J - H = K $ $G - I - J = \alpha \cdot K $	$I = K \cdot \beta, J = K , H = 0$ $G = K \cdot (1 + \beta + \alpha)$

En genel halde, verilen bir z domeni transfer fonksiyonuna ait katsayılar, E ve F devresine ait transfer fonksiyonlarının katsayılarıyla eşitlenerek devreye ilişkin kapasite değerleri hesaplanabilir. C , E veya C , F kapasiteleri payda polinomu tarafından belirlendikten sonra G , H , I , J kapasiteleri de pay polinomu tarafından belirlenir. G , H , I , J kapasitelerinin farklı kombinasyonları kullanılarak farklı filtre türleri gerçekleştirilebilir. Bu gerçeklemlere ilişkin tasarım denklemleri ve bu tasarımlara ait basit çözümler Tablo 2.1’de verilmiştir.

Bu sentez adımları sonucunda bulunan kapasite değerleri normalize olmayan değerlerdir. Şekil 2.9’da verilen devrede ilk işlemsel kuvvetlendiriciye ait çıkışın gerilim seviyesinin belirlenmesi gerekmektedir. Eğer bu gerilim çok büyük olursa ikinci işlemsel kuvvetlendirici aşırı yüklenir ve lineer olmayan çalışma bölgesine sürüklenir. Çok küçük olursa gürültüden kaynaklanan hatalar etkili olur.

$$T' \rightarrow \mu T' \quad (2.34a)$$

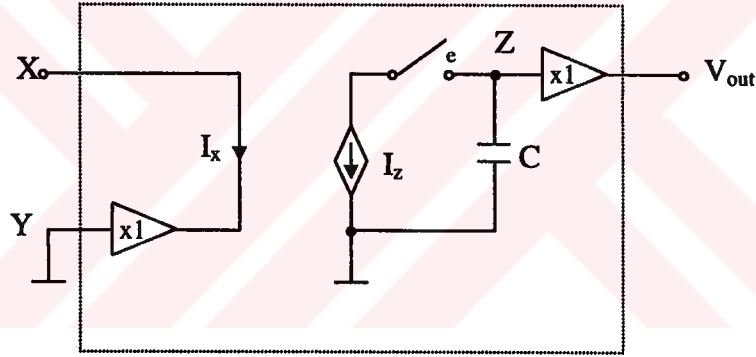
$$\{A, D\} \rightarrow \left\{ \frac{1}{\mu} \cdot A, \frac{1}{\mu} \cdot D \right\} \quad (2.34b)$$

V ile belirtilen çıkış geriliminin belirlenmesi için T' fonksiyonu μ katsayısı ile çarpılır. Bu yapılırken T fonksiyonunun değişmemesi istendiğinden sadece A ve D kapasiteleri (2.34b)’de görüldüğü gibi μ katsayısına bölünerek normalize edilir. Bu işlem T' fonksiyonunu μ ile çarpmaya eşdeğerdir. İkinci normalizasyon işlemi ise devredeki en küçük kapasitenin birim olması amacı ile yapılır. Böylece istenen maksimum kapasite oranları belirlenir. Daha önce de belirtildiği gibi (C, D, E, G, H) ve (A, B, I, J, F) olmak üzere iki kapasite grubu vardır ve bu gruplar birbirlerinde bağımsız olarak normalize edilirler. (C, D, E, G, H) ve (A, B, F, I, J) olarak belirlenen bu kapasiteler devre üzerinde incelendiğinde ilk grubun birinci işlemsel kuvvetlendiricinin, ikinci grubun ise ikinci işlemsel kuvvetlendiricinin girişine gelen kapasiteler olduğu anlaşılmaktadır[10].

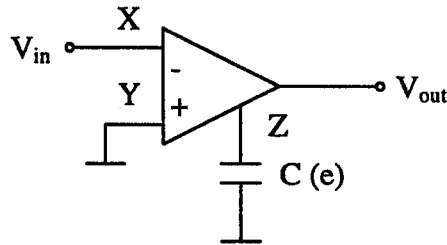
2.7 Anahtarlı-Kapasiteli Akım Taşıyıcı Eleman Kullanılarak Tasarlanan SC Filtre Devreleri

2.7.1 Anahtarlı-Kapasiteli Akım Taşıyıcı

Anahtarlı-kapasiteli akım taşıyıcı yapı bloğu Şekil 2.10'da verilmiştir. Bu yapı SCCC (switched-capacitor current conveyor) simgesi ile gösterilir. Yapı, bir adet y ucu topraklanmış pozitif akım kazançlı akım taşıyıcı, bu taşıyıcının çıkışına bağlanan kapasite ve yine akım taşıyıcı çıkışına bağlanan birim kazançlı bir buffer devresinden oluşur. Burada buffer devresinin kullanılmasındaki amaç, diğer katları sürmek için düşük çıkış empedansı sağlamak veya çıkıştan girişe geri besleme yapılabilme olanağını sağlamak şeklinde özetlenebilir.



Şekil 2.10 SCCC yapı bloğunun iç yapısı



Şekil 2.11 SCCC yapı bloğunun şematik gösterimi

Yapının X ucu görünen toprak olduğu için işlemsel kuvvetlendirici içeren aktif SC devrelerinde kullanılan anahtarlı kapasite yapıları burada da kullanılabilir. Yapı incelendiğinde her t anı için (2.35) eşitliklerinin yazılabileceği görülür.

$$V_x = V_y = 0 \quad (2.35a)$$

$$I_x = I_y \quad (2.35b)$$

$$V_{out} = V_z \quad (2.35c)$$

Devre incelendiğinde, tek saat fazı anında C kapasitesi üzerinden akım akmadığı dolayısıyla tek saat fazı anları için kapasite üzerindeki yükün sıfır olduğu görülür. O halde, çıkış düğümünün yük denklemi tek saat fazı anı için yazılırsa :

$$V_z^o C - z^{-1/2} V_z^e C = 0 \quad (2.36a)$$

$$V_z^o(z) = z^{-1/2} V_z^e \quad (2.36b)$$

eşitlikleri elde edilir. (2.36b) eşitliği, çıkış geriliminin tam saat periyodu boyunca tutulduğunu gösterir. Yapının çıkışının $t = nT$ örnekleme anlarında bir başka deyişle çift saat fazı anında örneklendiği kabul edildiğine göre Z düğümüne ilişkin düğün-yük denklemi (2.37)'de görüldüğü gibi yazılabilir.

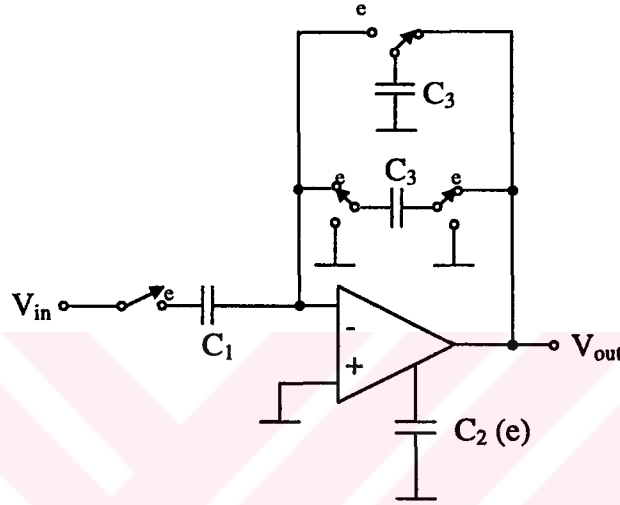
$$C_2 V_{out}(nT) - C_2 V_{out}\left(nT - \frac{T}{2}\right) = -I_z(nT) \cdot \frac{T}{2} \quad (2.37)$$

Bu denklemin z dönüşümü alınır ve (2.36b) eşitliği dikkate alınarak yeniden düzenlenirse (2.38) eşitliği elde edilir.

$$V_{out}^e(z) C(1 - z^{-1}) = -\Delta Q_z^e \quad (2.38)$$

Tanım bağıntıları ve z ucuna ait düğüm-yük denklemi bu şekilde elde edilen SCCC yapı bloğu, 1996 yılında Maundy ve Aronhime adlı iki kişi tarafından geliştirilmiştir[9]. Şimdi bu aktif eleman kullanılarak integratör, türev alıcı ve ikinci derece genel filtre devrelerinin nasıl gerçekleştirileceği incelenecektir.

2.7.2 SCCC Yapı Bloğu İle Oluşturulan İntegratör Devresi



Şekil 2.12 SCCC aktif elemanı ile oluşturulan integratör devresi

SCCC aktif eleman kullanılarak tasarlanan kayıplı eviren tip integratör devre yapısı Şekil 2.12'de görülmektedir. Analizi kolaylaştırmak amacıyla girişe uygulanan işaretin örneklenen değeri tam saat periyodu boyunca tutulmaktadır. Aynı zamanda SC devrenin, asıl periyodun %50'sini paylaşan ve üstüste gelmeyen Φ^o ve Φ^e saat fazlarını kullandığı varsayılmıştır. Devrenin girişinde görülen kapasitenin, üzerindeki gerilim tek saat fazı anında sabit kalır. Çünkü akım yolu mevcut olmadığından yüklenemez veya yükünü boşaltamaz. Devrenin X ucuna ait düğüm-yük denklemi çift saat fazı anı için (2.39) denkleminde görüldüğü gibi yazılabilir. Yukarıda yapılan saptamalar ve (2.36b) denklemi gözönüne alınır ve aşağıdaki denklem yeniden düzenlenirse (2.40) eşitliği elde edilir.

$$V_1^e C_1 - z^{-1/2} V_1^o C_1 + V_{out}^e C_3 + V_{out}^o z^{-1/2} C_3 = \Delta Q_X^e \quad (2.39)$$

$$V_1^e(1-z^{-1})C_1 + V_{out}^e C_3(1+z^{-1}) = \Delta Q_x^e \quad (2.40)$$

Devrede kullanılan akım taşıyıcı pozitif akım kazançlı olduğu için :

$$\Delta Q_x^e = \Delta Q_z^e \quad (2.41)$$

eşitliği yazılabilir. Yapının Z düğümüne ilişkin düğüm-yük denklemi (2.38)'de verilmiştir. Bu denklem, (2.41) ve (2.40) eşitliği birleştirilerek yeniden düzenlenirse

$$(1-z^{-1})V_{out}^e C_2 = -V_1^e C_1(1-z^{-1}) - V_{out}^e C_3(1-z^{-1}) \quad (2.42)$$

eşitliği elde edilir. Bu denklem V_{out}^e/V_{in}^e transfer fonksiyonunu verecek şekilde yeniden düzenlendiğinde (2.43)'de görülen transfer fonksiyonu elde edilir.

$$\frac{V_{out}^e}{V_{in}^e} = -\frac{\frac{C_1}{C_2+C_3}(1-z^{-1})}{1+\frac{C_3-C_2}{C_3+C_2}z^{-1}} = -\frac{K(1-z^{-1})}{1+b_1z^{-1}} \quad (2.43)$$

Elde edilen bu fonksiyon eviren türden kayıplı integratör yapısına ait transfer fonksiyonudur. Bu fonksiyonda görülen kapasite değerleri, iki fonksiyonun katsayıları birbirine eşitlenerek elde edilir. Bunun için gerekli tasarım eşitlikleri (2.44) eşitliklerinde görüldüğü gibi yazılabilir.

$$K = \frac{C_1}{C_2 + C_3} \quad (2.44a)$$

$$b_1 = \frac{C_3 - C_2}{C_3 + C_2} \quad (2.44b)$$

Bu eşitliklerde görülen C_3 kapasitesi bire eşit alınır ve eşitlikler yeniden düzenlenirse devreye ilişkin tasarım denklemleri :

$$C_3 = 1 \quad (2.45a)$$

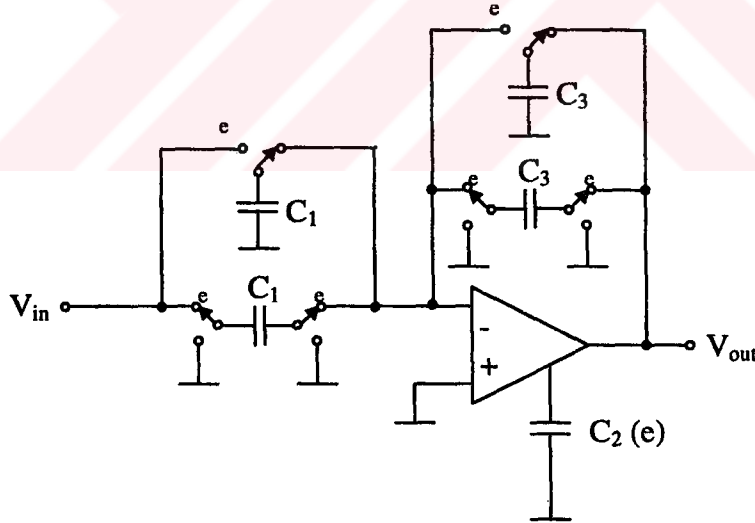
$$C_2 = \frac{1-b_1}{1+b_1} \quad (2.45b)$$

$$C_1 = (1+C_2) = \frac{K}{1+b_2} \quad (2.45c)$$

şeklinde elde edilir.

Paydada yer alan b_1 katsayısının mutlak değeri olarak daima birden küçük olacağı (2.44b) eşitliğinden açıkça görülmektedir. Bu da devrenin, tüm kararlı kutupları gerçekleyebileceğini gösterir. Eğer evirmeyen türden bir integratör tasarlanmak istenirse devrenin çıkışına bir evirici bağlanması yeterlidir.

2.7.3 SCCC Yapı Bloğu İle Oluşturulan Türev Alıcı Devresi



Şekil 2.13 SCCC yapı bloğu ile oluşturulan türev alma devresi

SCCC yapı bloğu ile oluşturulan türev alıcı devresi Şekil 2.13'de görülmektedir. İntegratör devresi için, analizi kolaylaştırmak amacıyla yapılan kabuller bu devre yapısı için de geçerlidir. Devrenin X düğümünü ait düğüm-yük denklemi çift saat fazı

anları için (2.46) eşitliğinde görüldüğü gibi yazılabilir.

$$V_1^e C_1 + z^{-1/2} V_1^o C_1 + V_{out}^e C_3 + V_{out}^o z^{-1/2} C_3 = \Delta Q_x^e \quad (2.46)$$

Daha önce yapılan kabuller ve (2.36) eşitliği gözönüne alınarak, (2.46) eşitliğinde görülen tek saat fazı anına ait gerilim değerleri yok edilir ve denklem yeniden düzenlenirse (2.47) eşitliği elde edilir.

$$V_1^e (1 + z^{-1}) C_1 + V_{out}^e C_3 (1 + z^{-1}) = \Delta Q_x^e \quad (2.47)$$

Yapıda kullanılan akım taşıyıcı pozitif akım kazançlıdır. Ayrıca yapının Z düğümüne ilişkin düğüm-yük denklemi (2.38) eşitliğinde verilmiştir. Bunlara göre (2.47) eşitliğinde görülen denklem, V_{out}^e / V_{in}^e transfer fonksiyonunu verecek şekilde yeniden düzenlenirse devrenin transfer fonksiyonu (2.48) eşitliğinde görüldüğü gibi elde edilir.

$$\frac{V_{out}^e}{V_{in}^e} = - \frac{\frac{C_1}{C_2 + C_3} (1 + z^{-1})}{1 - \frac{C_2 - C_3}{C_3 + C_2} z^{-1}} = - \frac{K(1 + z^{-1})}{1 - b_1 z^{-1}} \quad (2.48)$$

Elde edilen bu fonksiyonda görülen kapasitelerin değerlerini bulmak için transfer fonksiyonlarının katsayıları birbirine eşitlenmelidir. Bu işlem yapılır ve C_2 kapasitesi bire eşit alınırsa (2.49) eşitliklerinde görülen tasarım denklemleri elde edilir.

$$C_2 = 1 \quad (2.49a)$$

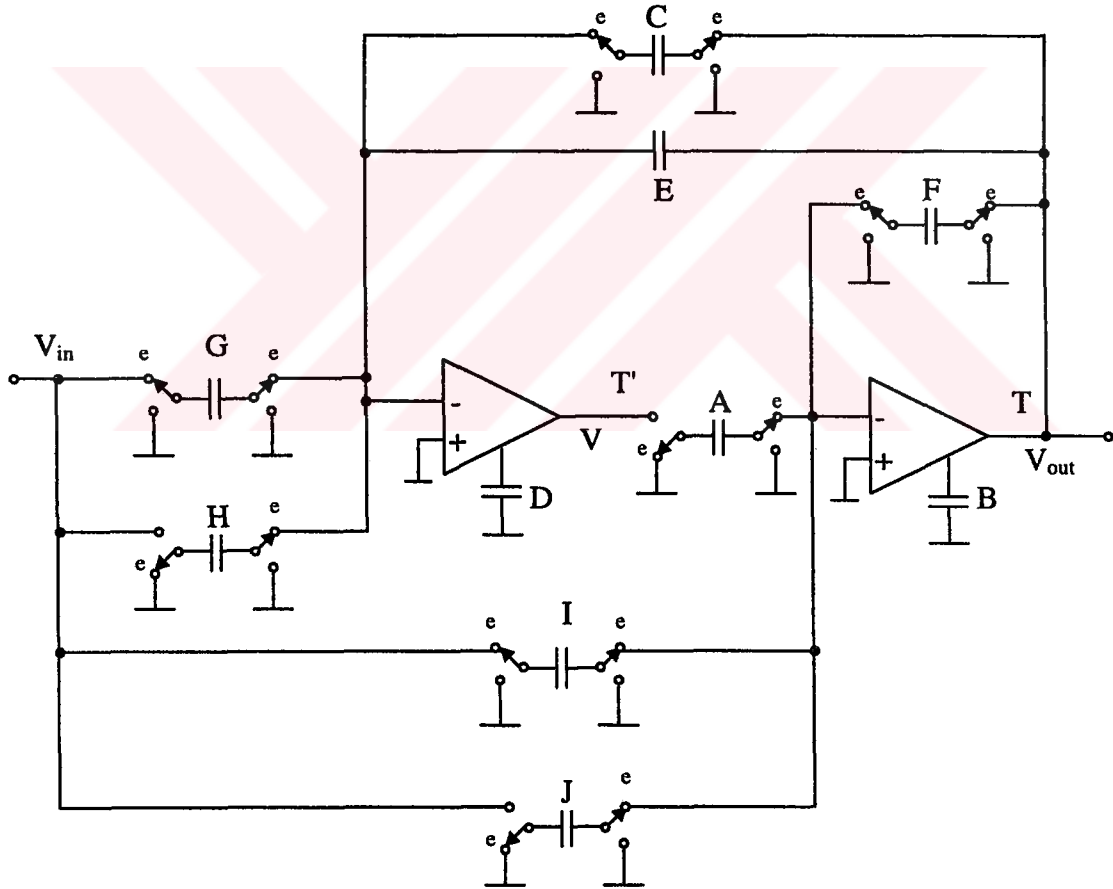
$$C_3 = \frac{1 - b_1}{1 + b_1} \quad (2.49b)$$

$$C_1 = (1 + C_3) = \frac{K}{1 + b_2} \quad (2.49c)$$

Elde edilen fonksiyondan da anlaşılabileceği gibi devre eviren tiptir. Evirmeyen tip türev alıcı gerçekleştirmek için bu yapının çıkışına işlemsel kuvvetlendirici kullanılarak hazırlanan birim kazançlı bir evirici bağlamak yeterlidir.

2.7.4 SCCC Yapı Bloğu İle Oluşturulan İkinci Derece SC Aktif Filtreler

Şekil 2.9'da işlemsel kuvvetlendirici kullanılarak tasarlanan SC aktif filtre devre topolojisi verilmişti. Bu devrede, küçük bir değişiklik yapmak şartıyla, işlemsel kuvvetlendirici yerine SCCC elemanı kullanılabilir. Bunun için Şekil 2.9'daki B ve D kapasiteleri SCCC'nin Z ucuna bağlanmalıdır. Bu yapıldığında elde edilen devre, ikinci dereceden SC aktif filtre transfer fonksiyonlarını gerçekleyebilir.(Şekil 2.14)



Şekil 2.14 SCCC yapı bloğu ile oluşturulan ikinci derece genel aktif SC filtre devresi

Devrenin girişine uygulanan işaretin örneklenen değeri tam saat periyodu boyunca tutulmaktadır. Bu devre de diğerleri gibi asıl periyodun %50'sini paylaşan ve üstüste gelmeyen Φ^e ve Φ^o saat fazlarını kullanmaktadır. Devrede yer alan SCCC'lerin x uçlarına ait düğüm yük denklemleri (2.50) eşitliklerinde görülmektedir.

$$V_{in}^e G - V_{in}^o H z^{-1/2} + V_{out}^e E - V_{out}^o z^{-1/2} E + V_{out}^e C = \Delta Q_{x1}^e \quad (2.50a)$$

$$V_{in}^e I - V_{in}^o J z^{-1/2} - V_{out}^o A z^{-1/2} = \Delta Q_{x2}^e \quad (2.50b)$$

Devrede kullanılan akım taşıyıcılar pozitif akım taşıyıcı olduklarından :

$$\Delta Q_{x1}^e = \Delta Q_{z1}^e \quad (2.51a)$$

$$\Delta Q_{x2}^e = \Delta Q_{z2}^e \quad (2.51b)$$

eşitlikleri yazılabilir. (2.38) eşitliği uyarınca Z düğümlerine ilişkin düğüm-yük denklemleri de (2.52) eşitliklerindeki gibi elde edilir.

$$V^e(z) B(1 - z^{-1}) = -\Delta Q_{z1}^e \quad (2.52a)$$

$$V_{out}^e(z) D(1 - z^{-1}) = -\Delta Q_{z2}^e \quad (2.52b)$$

Tüm bu eşitlikler T ve T' fonksiyonlarını verecek şekilde düzenlenirse :

$$T(z) = \frac{V_{out}^e}{V_{in}^e} = - \frac{DI + (AG - DI - DJ) \cdot z^{-1} + (DJ - AH) \cdot z^{-2}}{D \cdot (F + B) + (AC + AE - DF - 2BD) \cdot z^{-1} + (DB - AE) \cdot z^{-2}} \quad (2.53a)$$

$$T'(z) = \frac{V^e}{V_{in}^e} \quad (2.53b)$$

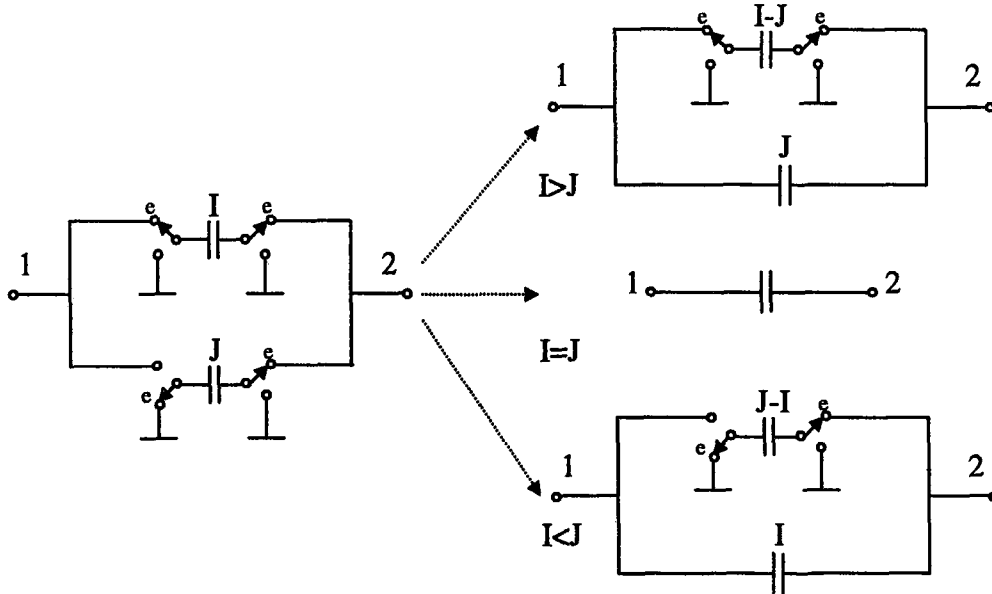
$$= \frac{(IC + IE - GF - GB) + (FH + BH + BG - JC - JE - IE) \cdot z^{-1} + (EJ - BH) \cdot z^{-2}}{D \cdot (F + B) + (AC + AE - DF - 2BD) \cdot z^{-1} + (DB - AE) \cdot z^{-2}}$$

eşitlikleri elde edilir. Bu eşitliklerde A, B, D kapasiteleri, ifadeleri basitleştirmek amacıyla bire eşit alınırsa (2.54)'deki eşitlikler elde edilir.

$$T(z) = \frac{V_{out}^e}{V_{in}^e} = - \frac{I + (G - I - J) \cdot z^{-1} + (J - H) \cdot z^{-2}}{(F + 1) + (C + E - F - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}} \quad (2.54a)$$

$$T'(z) = \frac{V^e}{V_{in}^e} = \frac{(IC + IE - GF - G) + (FH + H + G - JC - JE - IE) \cdot z^{-1} + (EJ - H) \cdot z^{-2}}{(F + 1) + (C + E - F - 2) \cdot z^{-1} + (1 - E) \cdot z^{-2}} \quad (2.54b)$$

Elde edilen bu fonksiyon, işlemsel kuvvetlendirici içeren devrelerin sağladığı transfer fonksiyonunun aynısıdır. Dolayısıyla o devre için yapılan bazı saptamalar bu devre için de geçerlidir. Bir başka deyişle bu devre de E ve F devresi olarak ikiye ayrılabilir. Bu devrenin de sıfırları G, H, I, J, kutupları C, F, E kapasiteleri tarafından belirlenir. Bu devrenin kararlılık bölgesi, işlemsel kuvvetlendirici içeren devreler için belirtilen kararlılık bölgesi ile aynıdır. Ayrıca bu devrenin birinci ve ikinci normalizasyon prosedürü ve birbirinden bağımsız olarak normalize edilebilen kapasite grupları, işlemsel kuvvetlendirici içeren yapının ki ile tamamen aynıdır.



Şekil 2.15 SC Eleman dönüşümleri

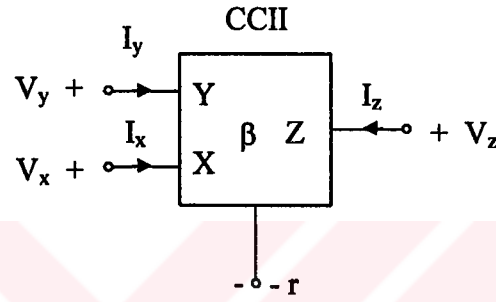
Gerek işlemsel kuvvetlendirici içeren, gerekse SCCC yapı bloğu içeren devrelerde (I,J) ve (G,H) kapasite çifti için farklı gerçeklemeler mümkündür. Bunun için Şekil 2.15’de gösterilen 1 nolu ucun gerilim kaynağına, 2 nolu ucun ise görünen toprak ucuna bağlanması gereklidir. Şekilden de görüldüğü gibi $I=J$ olduğunda anahtarlı iki kapasite yerine anahtarlanmamış tek bir kapasite gelir. Böylece devre sentezinde pek çok serbestlik ortaya çıkar. Ayrıca $I \neq J$ olması durumunda da bu dönüşüm kullanılabilmektedir. Böylece hem anahtar sayısından tasarruf edilmekte hem de toplam kapasite miktarı düşmektedir.



BÖLÜM 3

AKIM TAŞIYICI-SC AKTİF FİLTRELER

3.1 Akım Taşıyıcı Elemanı



Şekil 3.1 Akım taşıyıcı modeli

$$V_x = V_y \quad (3.1a)$$

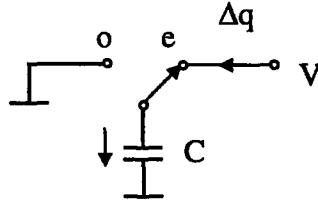
$$I_y = 0 \quad (3.1b)$$

$$I_z = \beta I_x \quad (3.1c)$$

$$\beta = \pm 1 \quad (3.1d)$$

bağıntıları ile tanımlanan dört uçlu elemana akım taşıyıcı denir ve Şekil 3.1'deki gibi gösterilir. Akım taşıyıcının x ve y uçlarına giriş, z ucuna çıkış, r ucuna da ortak uç denir. Bu nedenle I_x ve I_y akımı giriş akımı, I_z akımı da çıkış akımı olarak adlandırılmıştır. Her t anı için y ucuna ilişkin I_y sıfır olduğundan y ucu açık devre gibi davranır. β parametresi akım kazancı olarak adlandırılır. Bu parametre boyutsuz bir reel sayı olup, $\beta=-1$ ise akım taşıyıcıya negatif akım taşıyıcı, $\beta=1$ ise

görülmektedir.



Şekil 3.3 Lineer iletkenlik

Burada e anı için yük denklemi :

$$\Delta Q^e = CV^e \quad (3.3)$$

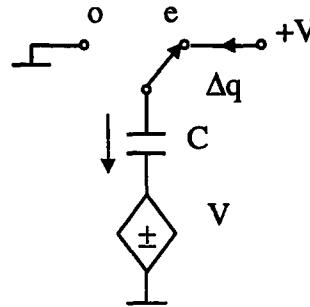
şeklinde yazılabilir. Aynı şekilde o anı için de (3.4)'de görülen :

$$\Delta Q^o = 0 \quad (3.4)$$

yük denklemi yazılabilir. Buradan da z domen transfer iletkenliği :

$$Y(z) = C \quad (3.5)$$

olarak bulunur [6].



Şekil 3.4 Evirmeyen gecikmeli iletkenlik

Evirmeyen gecikmeli iletkenlik olarak adlandırılan anahtarlı kapasitenin devre şeması Şekil 3.4'de görülmektedir. Burada e anı için yük denklemi :

$$\Delta Q^e = C(V^e - V^o) - z^{-1/2} C(0 - V^o) \quad (3.6a)$$

$$\Delta Q^e = z^{-1/2} C V^o \quad (3.6b)$$

biçiminde yazılabilir. Aynı şekilde o anı için yük denklemi :

$$\Delta Q^o = 0 \quad (3.7)$$

olarak yazılabilir. Saat periyodunun %50'si anahtarlar tarafından tek ve çift zaman aralıklarında paylaşıldığı varsayımı yapıldığı için :

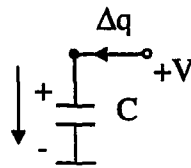
$$V^o = z^{-1/2} V^e \quad (3.8)$$

eşitliğini yazmak mümkündür. Bu denklem (3.6b) ifadesinde yerine konursa (3.9) ifadesi elde edilir.

$$\Delta Q^e = z^{-1} C V^e \quad (3.9)$$

Bu ifade düzenlenirse z domeni iletkenliği (3.10) eşitliğindeki gibi bulunur.

$$Y(z) = C z^{-1} \quad (3.10)$$



Şekil 3.5 Anahtarsız kapasite

Anahtarsız kapasite olarak da adlandırılan kompleks gecikmeli iletkenliğin devre şeması Şekil 3.5'de görülmektedir. Burada e anı için yük denklemi yazılacak olursa (3.11) eşitliği elde edilir.

$$\Delta Q^e = CV^e - (z^{-1/2}CV^o) \quad (3.11)$$

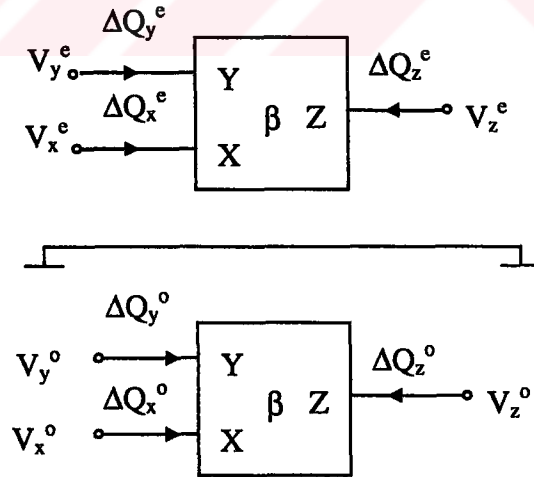
Saat periyodunun %50'si tek ve çift saat fazları tarafından paylaşıldığı kabulü yapıldığından (3.8) ifadesi, (3.11) ifadesinde yerine konur ve eşitlik yeniden düzenlenirse:

$$\Delta Q^e = C(1 - z^{-1})V^e \quad (3.12)$$

eşitliği elde edilir. Bu eşitlikten, anahtarsız kapasiteye ilişkin z domeni iletkenliği (3.13) eşitliğinde görüldüğü gibi bulunur [6].

$$Y(z) = C(1 - z^{-1}) \quad (3.13)$$

Akım taşıyıcı kullanılarak tasarlanan SC filtrelerin yapılarında içerdikleri akım taşıyıcıların z domeni eşdeğer devresi Şekil 3.6 da görülmektedir. Tanım bağıntıları ise (3.14) eşitliklerinde görüldüğü gibidir.



Şekil 3.6 Akım taşıyıcı z domeni eşdeğer devresi

$$\Delta Q_y^e = \Delta Q_y^o = 0 \quad (3.14a)$$

$$V_x^e = V_y^e \quad (3.14b)$$

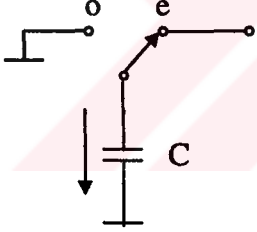
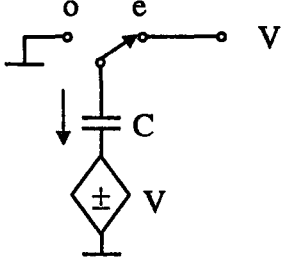
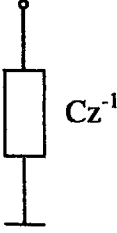
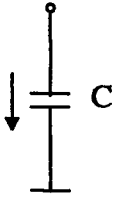
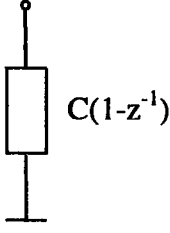
$$V_x^o = V_y^o \quad (3.14c)$$

$$\Delta Q_z^e = \beta \Delta Q_x^e \quad (3.14d)$$

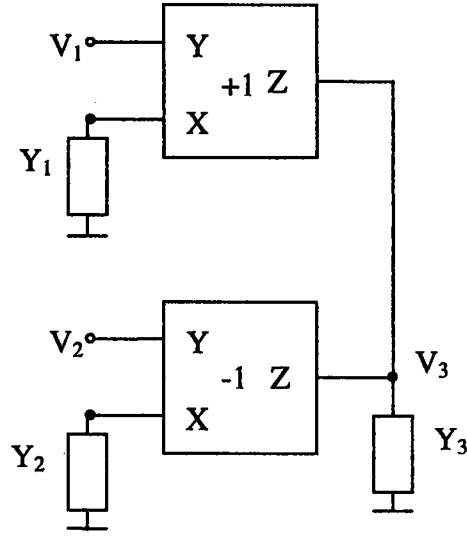
$$\Delta Q_z^o = \beta \Delta Q_x^o \quad (3.14e)$$

Akım taşıyıcı içeren SC aktif filtre yapılarında kullanılan anahtarlı kapasite yapıları ve bunlara ilişkin z domenî eşdeğer iletkenlikleri Tablo 3.1’de görölmektedir.

Tablo 3.1 Basit SC yapıları ve z domenî eşdeğerleri

Anahtarlı kapasiteler	Z domenî iletkenlikleri
	
	
	

3.3 CCII - SC Aktif Filtrelerin Sentezi



Şekil 3.7 Akım taşıyıcılar ile oluşturulan gerilim toplama devresi

Şekil 3.7’de akım taşıyıcı ile oluşturulmuş gerilim toplama devresi görülmektedir. Bu devre, akım taşıyıcıların tanım bağıntıları ve düğüm denklemleri yazılarak analiz edilirse (3.15) eşitlikleri elde edilir.

$$I_{x1} + V_1 Y_1 = 0 \quad (3.15a)$$

$$I_{x2} + V_2 Y_2 = 0 \quad (3.15b)$$

$$I_{z1} + I_{z2} + V_3 Y_3 = 0 \quad (3.15c)$$

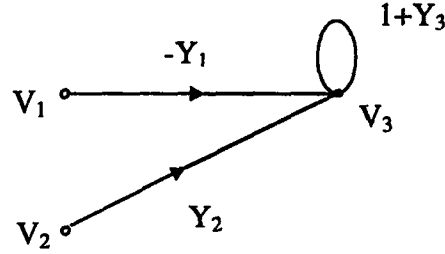
$$I_{z1} = +I_{x1} \quad (3.15d)$$

$$I_{z2} = -I_{x2} \quad (3.15e)$$

Bu denklemler düzenlendiğinde (3.16) denklemi elde edilir. Bu denklemin sağ ve sol tarafına V_3 değeri eklenir ve işaret akış diyagramı çizilirse Şekil 3.8’de görülen

işaret akış diyagramı bulunur.

$$-V_1 Y_1 + V_2 Y_2 + V_3 Y_3 = 0 \quad (3.16)$$



Şekil 3.8 Şekil 3.7'deki devrenin işaret akış diyagramı

Eğer z domenli transfer fonksiyonu, Şekil 3.8'de görülen işaret akış diyagramını alt işaret akış diyagramı olarak kabul eden ve bunların birleşiminden oluşan bir işaret akış diyagramı ile temsil edilebiliyorsa; bu fonksiyon, Şekil 3.7'de görülen devreyi alt devre olarak kabul eden ve bunların birleşiminden oluşan bir devre topolojisi ile gerçekleştirilebilir. Bu sentez metodunda dal kazançları, fonksiyonun katsayıları cinsinden ifade edilir. Bu dal kazançlarının mutlak değerleri, devrede yer alan elemanların z domenli iletkenlik değerlerini belirler. Dal kazançlarının işaretleri de elemanların bağlanacağı akım taşıyıcıların akım kazançlarını belirler. Bu devrede bir önceki bölümde tanımlanan anahtarlı kapasite yapıları, x ucu ile toprak ve z ucu ile toprak arasına bağlanır. Devrenin girişi y olup girişe uygulanan işaret tam saat periyotlu bir işarettir [7,12].

3.4 Birinci Dereceden CCII-SC Aktif Filtreler

3.4.1 İntegratör Yapısı

Genel olarak birinci dereceden bir filtrenin transfer fonksiyonu (3.17) ifadesinde görülmektedir. Bu transfer fonksiyonunu CCII-SC aktif filtreler ile gerçeklemek için, öncelikle bu transfer fonksiyonu bilineer dönüşüm metodu ile z domenine çevrilmelidir. Bunun için s yerine (3.18)'de görülen ifade yerleştirilmelidir. Bu yapıldığında (3.19)'da görülen transfer fonksiyonu elde edilir.

$$T(s) = \frac{p}{s+p} \quad (3.17)$$

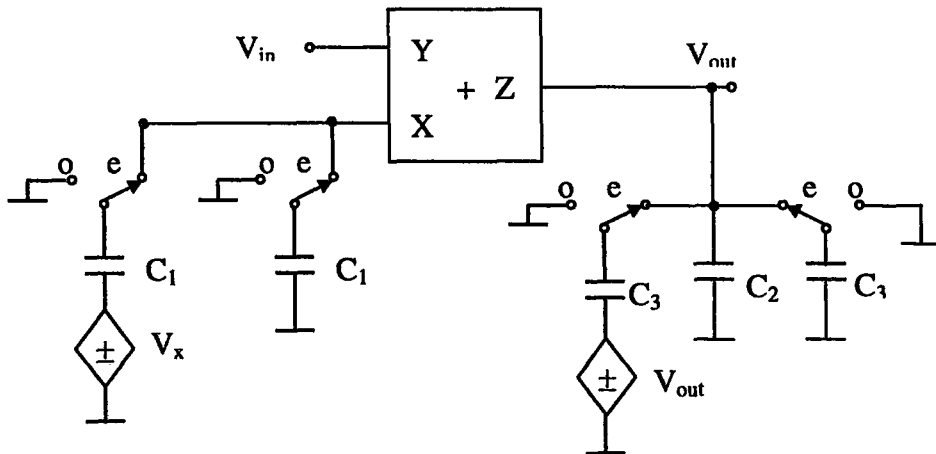
$$s \rightarrow \frac{2}{\tau} \frac{1-z^{-1}}{1+z^{-1}} \quad (3.18)$$

Burada τ örnekleme süresidir.

$$T(z) = \frac{\frac{p\tau}{2+p\tau}(1+z^{-1})}{1 + \frac{p\tau-2}{p\tau+2}z^{-1}} \quad (3.19a)$$

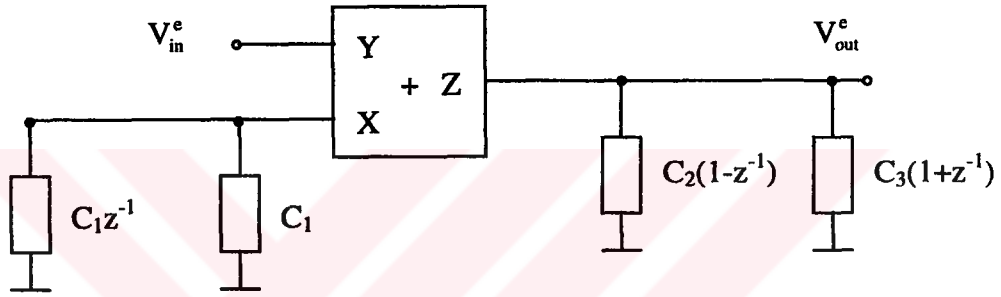
$$T(z) = \frac{K(1+z^{-1})}{1+b_1z^{-1}} \quad (3.19b)$$

(3.19b) ifadesinde görülen $T(z)$ transfer fonksiyonunun bir filtre devresi ile gerçekleştirilebilmesi için kararlı olması şarttır. Kararlılık için gerek ve yeter şart, fonksiyonun kutuplarının z düzleminde birim daire içinde olmasıdır. Bu şartın sağlanması için b_1 katsayısının mutlak değeri birden küçük olmalıdır.

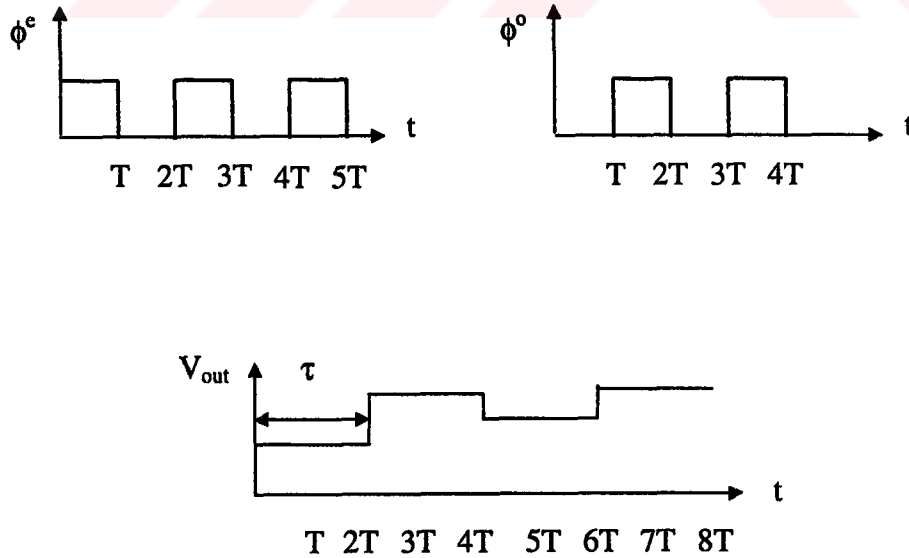


Şekil 3.9 (a) Birinci dereceden CCII-SC aktif filtre devre topolojisi

Bu biçimde elde edilen kararlı z domeni transfer fonksiyonu Şekil 3.9'da görülen birinci derece CCII-SC aktif filtre yapısı ile gerçekleştirilebilir. Bu yapıya ait transfer fonksiyonunu elde etmek için bazı şartların belirlenmesi gerekir. İlk olarak analizde kolaylık sağlaması bakımından Şekil 3.9'da görülen SC devrenin; asıl periyodun %50'sini paylaşan ve üst üste gelmeyen ϕ^e ve ϕ^o saat fazlarını kullandıkları kabul edilmektedir. Giriş işaretinin örneklenen değerinin de tam saat periyodu boyunca tutulduğu varsayılmaktadır. Saat fazlarına ve giriş dalga formuna ait bir örnek Şekil 3.10'da görülmektedir. Gerçekte SC devrelerin anahtarlama fazları bu girişi işleyecek şekilde seçilmiştir.



Şekil 3.9 (b) Birinci dereceden CCII-SC aktif filtrenin z domeni eşdeğer devresi



Şekil 3.10 Saat ve giriş dalga formu

$$V_{in}^o = z^{-1/2} V_{in}^e \quad (3.20)$$

Bu şartlar dikkate alındığında (3.20) ifadesi yazılabilir. Şekil 3.9'da görülen devrenin tanım bağıntıları ve akım taşıyıcının x ucuna ait yük denklemi dikkate alınırsa tek saat fazı anları için (3.21) eşitlikleri yazılabilir. Akım taşıyıcının z ucuna ait yük denklemi de yine tek saat fazı anları için (3.22)'deki gibi yazılabilir.

$$\Delta Q_y^o = 0 \quad (3.21a)$$

$$\Delta Q_x^o = 0 \quad (3.21b)$$

$$\Delta Q_z^o = \Delta Q_x^o \quad (3.21c)$$

$$\Delta Q_z^o + C_2 V_{out}^o - C_2 z^{-1/2} V_{out}^e = 0 \quad (3.22)$$

Bu denklem, (3.21c) eşitliği dikkate alınarak yeniden düzenlenirse (3.23) eşitliği elde edilir.

$$V_{out}^o = z^{-1/2} V_{out}^e \quad (3.23)$$

Devrenin hem girişi hem de çıkışı, tam saat periyodu boyunca tutulduğundan devrenin analizi için yazılan denklemlerde sadece çift saat fazı gözönüne alınmaktadır. Buradan hareketle devrenin x ve z ucuna ait düğüm yük denklemleri (3.24) eşitliklerindeki gibi yazılabilir.

$$C_1 V_x^e + [(V_x^e - V_x^e) C_1 - z^{-1/2} (0 - V_x^o) C_1] + \Delta Q_x^e = 0 \quad (3.24a)$$

$$C_3 V_{out}^e + [(V_{out}^e - V_{out}^e) C_3 - z^{-1/2} (0 - V_{out}^o) C_3] + C_2 V_{out}^e - z^{-1/2} C_2 V_{out}^o + \Delta Q_z^e = 0 \quad (3.24b)$$

(3.20) ve (3.23) eşitlikleri, elde edilen (3.24) eşitliklerinde yerine konur ve gerekli düzenlemeler yapılırsa (3.25) eşitlikleri elde edilir.

$$C_1(1+z^{-1})V_x^e + \Delta Q_x^e = 0 \quad (3.25a)$$

$$C_2(1-z^{-1})V_{out}^e + C_3(1+z^{-1})V_{out}^e + \Delta Q_z^e = 0 \quad (3.25b)$$

Akım taşıyıcının tanım bağıntıları uyarınca (3.26) eşitliklerini yazmak mümkündür.

$$V_x^e = V_y^e = V_{in}^e \quad (3.26a)$$

$$\Delta Q_z^e = \Delta Q_x^e \quad (3.26b)$$

Bu eşitlikler (3.25a) ve (3.25b) eşitliklerinde yerine konur ve denklemler düzenlenirse (3.27)'de görülen transfer fonksiyonu elde edilir.

$$\frac{V_{out}^e}{V_{in}^e} = \frac{\frac{C_1}{C_2+C_3}(1+z^{-1})}{1 + \frac{C_3-C_2}{C_3+C_2}z^{-1}} \quad (3.27)$$

Bu transfer fonksiyonun katsayıları (3.18b)'deki transfer fonksiyonun katsayıları ile eşitlendiğinde, K ve b_1 katsayıları (3.28)'de eşitliklerindeki gibi elde edilir. Bu eşitlikten de görülebileceği gibi b_1 katsayısı daima 1'den küçüktür; bir başka deyişle elde edilen transfer fonksiyonu daima kararlılık koşulunu sağlamaktadır.

$$K = \frac{C_1}{C_2+C_3} \quad (3.28a)$$

$$b_1 = \frac{C_3-C_2}{C_3+C_2} \quad (3.28b)$$

(3.28) eşitliklerinde görülen C_3 kapasitesinin değeri 1 alınır ve devreye ilişkin tasarım eşitlikleri yazılırsa (3.29) eşitlikleri elde edilir. Bu tasarım eşitliklerinden elde edilen, normalize edilmemiş kapasite değerleri bir katsayı ile çarpılarak normalize edilmelidirler.

$$C_3 = 1 \quad (3.29a)$$

$$C_2 = \frac{1-b_1}{1+b_1} \quad (3.29b)$$

$$C_1 = K(1+C_2) \quad (3.29c)$$

Şekil 3.9'da görülen birinci dereceden CCII-SC aktif filtre devresi evirmeyen türden kayıplı integratör olarak da adlandırılır. Yapıda kullanılan ve lineer iletkenlik olarak isimlendirilen C_3 anahtarlı kapasite yapısı devrenin kayıplı integratör olmasını sağlamaktadır. Yine yapıda kullanılan ve gecikmeli iletkenlik olarak isimlendirilen C_3 anahtarlı kapasite yapısı (3.19b) eşitliğinde görülen b_1 katsayısının, iki kapasite farkı cinsinden ifade edilmesini sağlar. Bu eşitlik (3.28b)'de görülmektedir. Böylece -1 ile 1 arasında değişen b_1 katsayısının negatif olması halinde de fonksiyon, bu devre yapısı ile gerçekleştirilebilir. Eğer eviren türden integratör tasarlanmak istenirse, akım taşıyıcının akım kazancını değiştirmek bir başka deyişle negatif akım taşıyıcı kullanmak yeterlidir. Devrenin gerçekleştirilmesi aşamasında parazitik kapasitelerin etkisini minimum düzeye indirmek için devrede kullanılan minimum kapasite değeri 5-10pf olmalıdır.

3.4.2 Türev Alıcı

Türev alıcı z domeni transfer fonksiyonu, integratör devresine ait transfer fonksiyonunda z yerine $-z$ konarak elde edilebilir. Bu şekilde elde edilen türev alıcı devreye ilişkin transfer fonksiyonu (3.30) eşitliğinde verilmiştir.

$$T(z) = \frac{K(1 - z^{-1})}{1 - b_1 z^{-1}} \quad (3.30)$$

Bu transfer fonksiyonunu gerçekleyen CCII-SC türev alıcı devre yapısı Şekil 3.11a' da görülmektedir. Bu yapının z domeni eşdeğer devresi de Şekil 3.11b'de görülmektedir. Şekil 3.11a'da görülen devrenin x ve z uçlarına ait düğüm-yük denklemleri, integratör devresinde olduğu gibi yazılır ve gerekli ara işlemler yapılırsa bu devreye ilişkin transfer fonksiyonu (3.31)'de görüldüğü gibi elde edilir.

$$\frac{V_{out}^e}{V_{in}^e} = \frac{\frac{C_1}{C_2 + C_3} (1 - z^{-1})}{1 - \frac{C_3 - C_2}{C_3 + C_2} z^{-1}} \quad (3.31)$$

Bu transfer fonksiyonunu katsayıları (3.30) eşitliğindeki katsayılar ile eşitlenirse K ve b_1 katsayıları (3.32) eşitliklerindeki gibi elde edilirler.

$$K = \frac{C_1}{C_2 + C_3} \quad (3.32a)$$

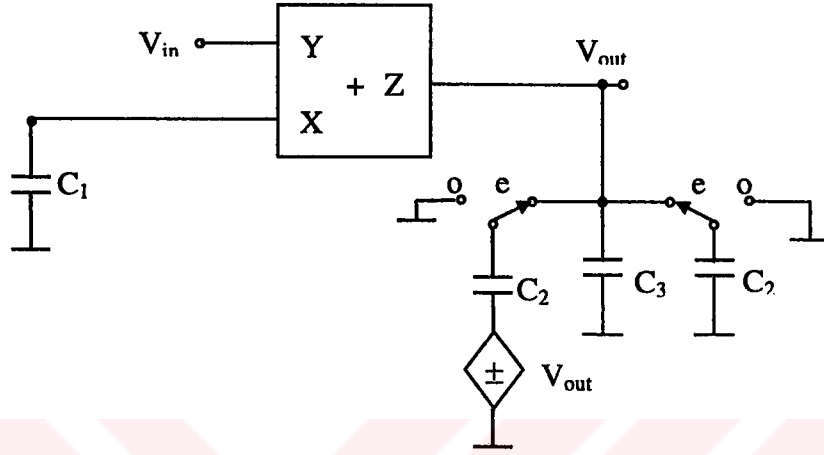
$$b_1 = \frac{C_3 - C_2}{C_3 + C_2} \quad (3.32b)$$

(3.32) eşitliklerinde görülen C_2 kapasitesi 1 alınarak devreye ilişkin tasarım eşitlikleri yazılırsa (3.33) eşitlikleri elde edilir. Bu tasarım eşitliklerinden elde edilen normalize edilmemiş kapasite değerleri, bir katsayı ile çarpılarak istenen değerlere çekilmek suretiyle normalize edilebilirler. Eğer eviren türden türev alıcı tasarlanmak istenirse akım taşıyıcının akım kazancını değiştirmek bir başka deyişle negatif akım taşıyıcı kullanmak yeterlidir.

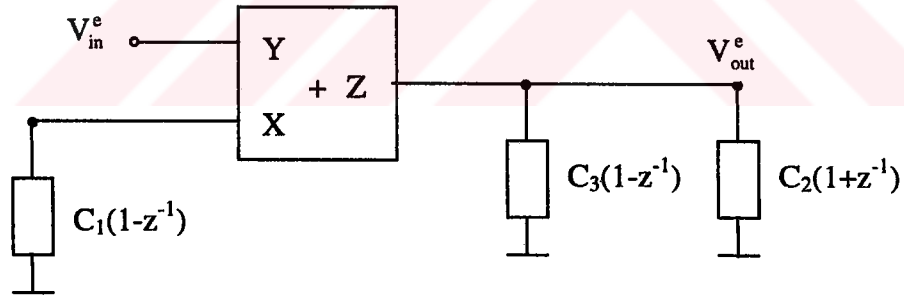
$$C_2 = 1 \quad (3.33a)$$

$$C_3 = \frac{1+b_1}{1-b_1} \quad (3.33b)$$

$$C_1 = K(1+C_3) \quad (3.33c)$$



Şekil 11(a) Birinci dereceden türev alıcı CCII-SC aktif filtre devre topolojisi



Şekil 3.11 (b) Şekil 3.11 (a)'daki devrenin z domeni eşdeğer devresi

3.5 İkinci Dereceden CCII-SC Aktif Filtreler

3.5.1 İkinci Dereceden CCII-SC Aktif Filtrelerin Genel Yapısı

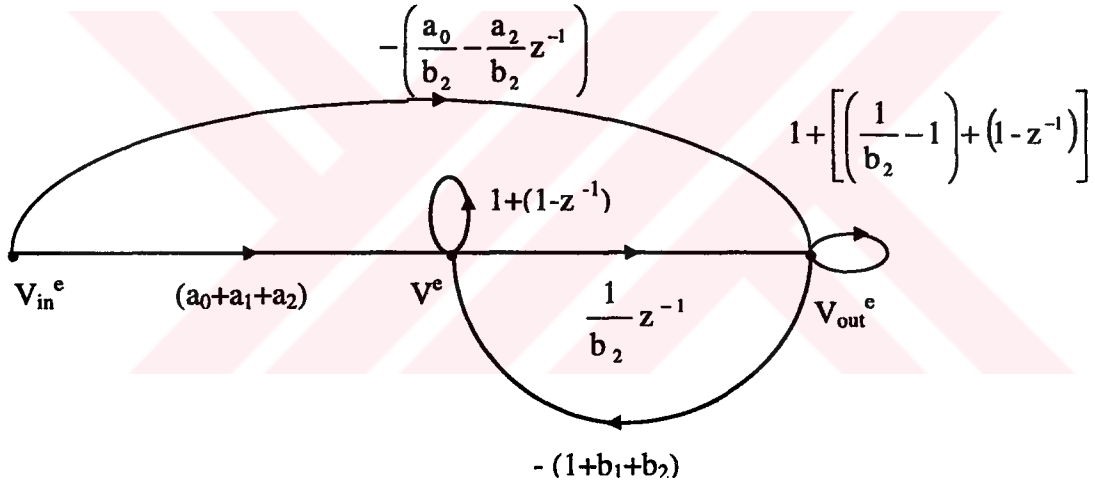
Genel olarak ikinci dereceden bir filtrenin z domeni transfer fonksiyonu (3.34)'de görülmektedir.

$$T(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (3.34)$$

Bu transfer fonksiyonunun bir filtre devresiyle gerçekleştirilmesi için fonksiyonun kararlı olması şarttır. Fonksiyonun kararlı olması için tüm kutuplarının z düzleminde birim dairenin içinde olması bir başka değişle $|z| < 1$ olması gerekir. Bunun için (3.35) eşitlikleri sağlanmalıdır.

$$|b_2| < 1 \quad (3.35a)$$

$$1 + b_2 > \pm b_1 \quad (3.35b)$$



Şekil 3.12 (a) Genel ikinci dereceden z domeni aktif filtre transfer fonksiyonun işaret akış diyagramı

Bu biçimde elde edilen kararlı z domeni transfer fonksiyonlarının Şekil 3.12(a)'da görülen işaret akış diyagramı ile temsil edilebildiği mason kazanç formülü kullanılarak gösterilebilir. Bu işaret akış diyagramı, Şekil 3.12(b)'deki işaret akış diyagramının aynısıdır.

Şekil 3.12(a) ve Şekil 3.12(b)'de görülen işaret akış diyagramlarının dal kazançları birbirlerine eşitlenirse iletkenlik değerleri (3.36) eşitliklerinde görüldüğü gibi elde edilir.

$$Y_1 = a_0 + a_1 + a_2 \quad (3.36a)$$

$$Y_2 = \frac{a_0}{b_2} \quad (3.36b)$$

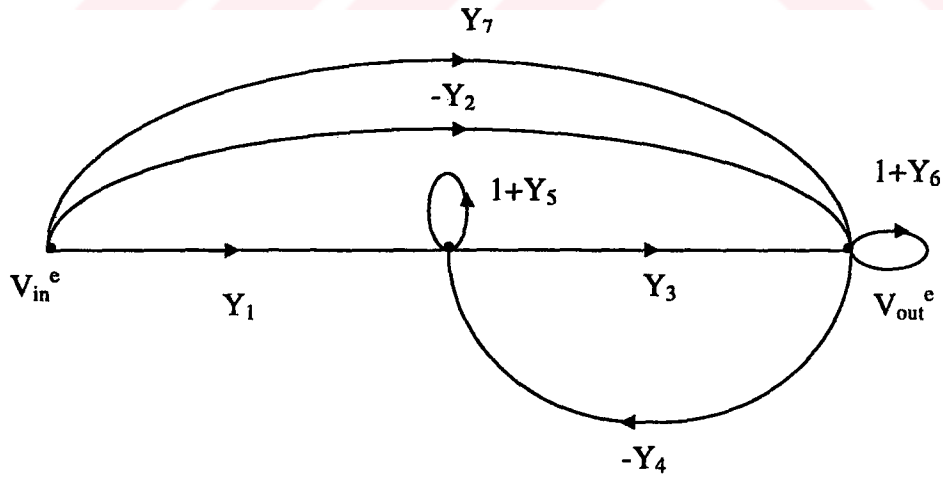
$$Y_3 = \frac{1}{b_2} z^{-1} \quad (3.36c)$$

$$Y_4 = 1 + b_1 + b_2 \quad (3.36d)$$

$$Y_5 = 1 - z^{-1} \quad (3.36e)$$

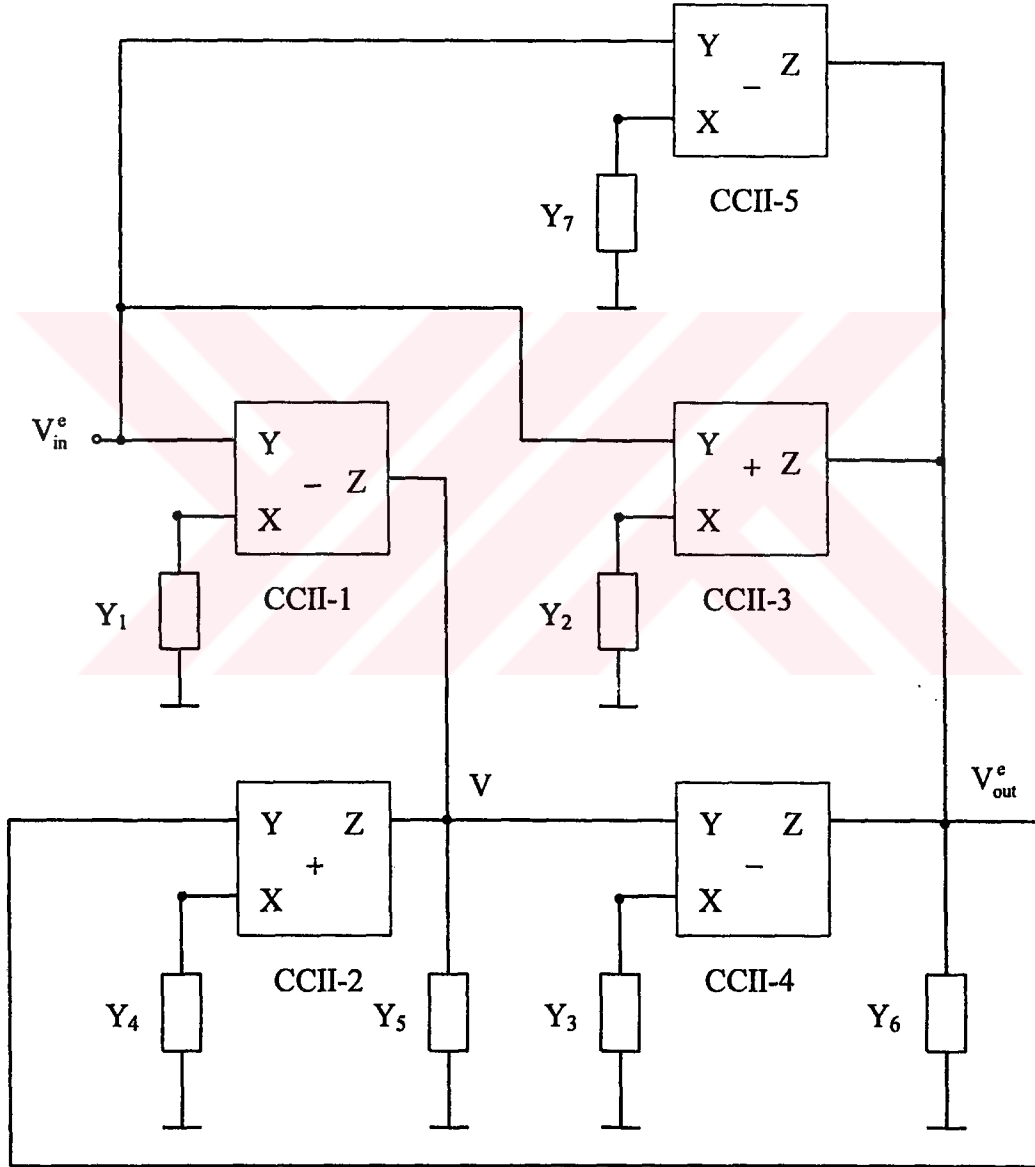
$$Y_6 = \left(\frac{1}{b_2} - 1 \right) + (1 - z^{-1}) \quad (3.36f)$$

$$Y_7 = \frac{a_2}{b_2} z^{-1} \quad (3.36g)$$



Şekil 3.12(b) Genel ikinci dereceden z domeni transfer fonksiyonunun işaret akış diyagramının farklı bir versiyonu

O halde CCII-SC aktif filtrelerin sentezi yöntemine dayanarak ve Şekil 3.12(b)'de görülen işaret akış diyagramı dikkate alınarak Şekil 3.13(a)'da görülen genel CCII-SC aktif filtrenin z domeni eşdeğer devresi çizilebilir. Elde edilen bu z domeni eşdeğer devrede yer alan z domeni iletkenliklerinin yerine, Tablo 3.1'de verilen karşılıkları yani anahtarlı kapasite yapıları yerleştirilirse Şekil 3.13(b)'de görülen ikinci dereceden genel CCII-SC aktif filtre devre topolojisi elde edilir.



Şekil 3.13(a) İkinci dereceden genel CCII-SC aktif filtre z domeni eşdeğer devre yapısı

periyodun %50'sini paylaşan ve üst üste gelmeyen ϕ_e ve ϕ_o saat fazlarını kullandıkları kabul edilmektedir. Giriş işaretinin örneklenen değeri yine tam saat periyodu boyunca tutulmaktadır. O halde bu devre yapısı için (3.37) eşitliğini yazmak mümkündür.

$$V_{in}^o = z^{-1/2} V_{in}^e \quad (3.37)$$

Birinci ve ikinci akım taşıyıcıya ilişkin tanım bağıntıları ve düğüm yük denklemleri tek saat fazı anı için yazılacak olursa (3.38) eşitlikleri elde edilir.

$$\Delta Q_{x1}^o = 0 \quad (3.38a)$$

$$\Delta Q_{x2}^o = 0 \quad (3.38b)$$

$$\Delta Q_{z1}^o = -\Delta Q_{x1}^o = 0 \quad (3.38c)$$

$$\Delta Q_{z2}^o = \Delta Q_{x2}^o = 0 \quad (3.38d)$$

$$\Delta Q_{z1}^o + \Delta Q_{z2}^o + (CV^o - Cz^{-1/2} V^e) = 0 \quad (3.38e)$$

Bu eşitlikler düzenlenirse (3.39) eşitliği elde edilecektir.

$$V^o = z^{-1/2} V^e \quad (3.39)$$

Bu işlemlerin aynısı üçüncü, dördüncü, beşinci akım taşıyıcılar için yapıldığında (3.40) eşitliği elde edilecektir.

$$V_{out}^o = z^{-1/2} V_{out}^e \quad (3.40)$$

O halde devrenin çıkışının da girişi gibi tam saat periyodu boyunca tutulduğu ve sadece çift saat fazlarında değiştiği söylenebilir. Buna göre devrenin analizi için

yazılan denklemlerde sadece çift saat fazı gözönüne alınmaktadır.

Şekil 3.13(b)'de görülen ikinci dereceden genel CCII-SC aktif filtrenin transfer fonksiyonunu bulmak için akım taşıyıcıların tanım bağıntılarından ve akım taşıyıcıların x ve z uçlarına ait düğüm yük denklemlerinden yararlanılır. Buna göre, akım taşıyıcıların tanım bağıntıları uyarınca (3.41) eşitlikleri yazılabilir.

$$V_{x1} = V_{y1} = V_{in} \quad (3.41a)$$

$$V_{x2} = V_{y2} = V_{out} \quad (3.41b)$$

$$V_{x3} = V_{y3} = V_{in} \quad (3.41c)$$

$$V_{x4} = V_{y4} = V_{in} \quad (3.41d)$$

$$V_{x5} = V_{y5} = V \quad (3.41e)$$

$$\Delta Q_{z1}^e = -\Delta Q_{x1}^e \quad (3.41f)$$

$$\Delta Q_{z2}^e = \Delta Q_{x2}^e \quad (3.41g)$$

$$\Delta Q_{z3}^e = \Delta Q_{x3}^e \quad (3.41h)$$

$$\Delta Q_{z4}^e = -\Delta Q_{x4}^e \quad (3.41i)$$

$$\Delta Q_{z5}^e = -\Delta Q_{x5}^e \quad (3.41i)$$

Birinci ve ikinci akım taşıyıcıların x uçlarına ait düğüm yük denklemleri (3.42) eşitliklerinde görülmektedir.

$$\Delta Q_{x1}^e + AV_{in}^e = 0 \quad (3.42a)$$

$$\Delta Q_{x2}^e + BV_{out}^e = 0 \quad (3.42b)$$

Aynı akım taşıyıcıların ortak z ucuna ait düğüm yük denklemi, (3.43)'de görüldüğü gibi yazılabilir. Bu denklem, (3.39), (3.41), ve (3.42) eşitlikleri gözönüne

$$\Delta Q_{z1}^e + \Delta Q_{z1}^e + C(V^e - z^{-1/2}V^o) = 0 \quad (3.43)$$

alınarak yeniden düzenlenirse (3.44)'de görülen eşitlik elde edilir. Üçüncü, dördüncü

$$AV_{in}^e - BV_{out}^e + C(1 - z^{-1})V^e = 0 \quad (3.44)$$

ve beşinci akım taşıyıcıların x uçlarına ait düğüm yük denklemleri de (3.45) eşitliklerinde görülmektedir.

$$\Delta Q_{x3}^e + GV^e = 0 \quad (3.45a)$$

$$\Delta Q_{x4}^e + Dz^{-1/2}V^o = 0 \quad (3.45b)$$

$$\Delta Q_{x5}^e + Hz^{-1/2}V_{in}^o = 0 \quad (3.45c)$$

Aynı akım taşıyıcıların ortak z ucuna ait düğüm yük denklemi (3.46)'da görüldüğü gibi yazılabilir.

$$\Delta Q_{z5}^e + \Delta Q_{z4}^e + \Delta Q_{z3}^e + E(V_{out}^e - z^{-1/2}V_{out}^o) + FV_{out}^e = 0 \quad (3.46)$$

Bu denklem (3.37), (3.39), (3.40), (3.41), (3.45) eşitlikleri gözönüne alınarak yeniden düzenlenirse (3.47)'de görülen eşitlik elde edilir.

$$-(G + Hz^{-1})V_{in}^e + Dz^{-1}V^e + E(1 - z^{-1})V_{out}^e + FV_{out}^e = 0 \quad (3.47)$$

(3.44) ve (3.47) eşitlikleri, T ve T' fonksiyonlarını bir başka deyişle, sırasıyla V_{out}^e/V_{in}^e , V^e/V_{in}^e gerilim transfer fonksiyonlarını verecek şekilde düzenlenirse (3.48) ve (3.49) eşitlikleri bulunur.

$$T(z) = \frac{GC + (DA - GC - HC)z^{-1} + HCz^{-2}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.48)$$

$$T'(z) = \frac{(BG - AE - AF) + (BH - AE)z^{-1}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.49)$$

(3.48) ve (3.49) eşitliklerinde görülen A, B, C, D, E, F, G, H kapasitelerinin değerleri (3.50) eşitliklerinde görülmektedir.

$$A = a_0 + a_1 + a_2 \quad (3.50a)$$

$$B = 1 + b_1 + b_2 \quad (3.50b)$$

$$C = 1 \quad (3.50c)$$

$$D = \frac{1}{b_2} \quad (3.50d)$$

$$E = 1 \quad (3.50e)$$

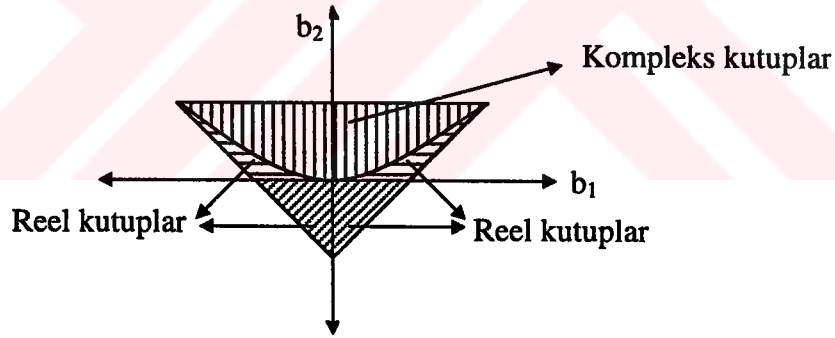
$$F = \frac{1}{b_2} - 1 \quad (3.50f)$$

$$G = \frac{a_0}{b_2} \quad (3.50g)$$

$$H = \frac{a_2}{b_2} \quad (3.50h)$$

(3.50) eşitliklerinden görüldüğü gibi devrede yer alan kapasite değerleri fonksiyonun katsayıları cinsinden elde edilmiştir. Eğer bu katsayılardan biri veya birkaçı negatif olursa bunların belirlediği kapasiteler negatif değer alamazlar. Böyle bir durumda, bu katsayıların mutlak değerleri ilgili kapasite değerlerini belirler. Bu katsayılardan ileri gelen negatiflik, bu kapasitelerin bağlı olduğu akım taşıyıcıların akım kazançları -1 ile çarpılmak suretiyle kompanze edilmelidir. Örneğin bu kapasitenin bağlı olduğu akım taşıyıcı pozitif akım kazançlı ise bunun yerine negatif akım kazançlı akım taşıyıcı kullanılmalıdır. Böylece hem kapasitelerin değeri pozitif olur hem de istenen transfer fonksiyonu sağlanır.

Kutupların yerleşiminde kararlılığın sağlanması zorunludur. Bu amaçla, daha önce (3.35) eşitsizliklerinde kararlılık koşulu verilen b_1 ve b_2 katsayılarına ilişkin kutup yerleşim üçgeninden yararlanılmaktadır. (3.35) eşitsizliklerinden yararlanılarak çizilen bu üçgen Şekil 3.14'de görülmektedir. Bu üçgenin üst kısmında kalan parabolik alan kararlı kompleks kutuplara karşı düşen, alt kısmında kalan bölge de kararlı reel kutuplara karşı düşen b_1 ve b_2 parametrelerini gösterir.



Şekil 3.14 $D(z) = 1 + b_1 z^{-1} + b_2 z^{-2}$ için kararlı kutup yerleşim üçgeni

Devreye ilişkin kararlılık koşulunu elde edebilmek için tasarım eşitliklerinden yararlanılmaktadır. Kapasite değerlerinin pozitif reel sayı olacağı gözönünde bulundurularak bu tasarım eşitlikleri yazılırsa (3.51) eşitlikleri elde edilir.

$$b_2 = \frac{EC}{EC + FC} \quad (3.51a)$$

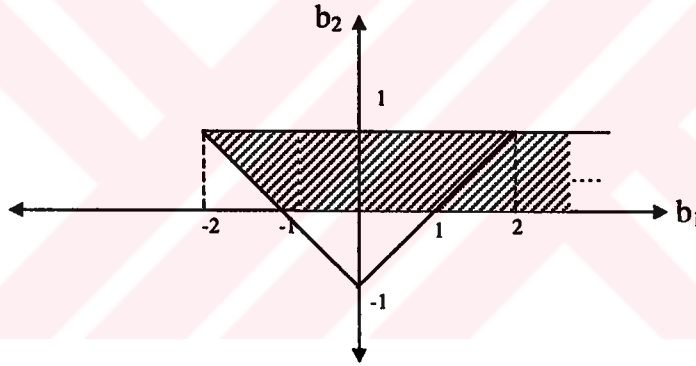
$$b_1 = \frac{DB - 2EC - FC}{EC + FC} \quad (3.51b)$$

Buradan b_1 ve b_2 parametreleri için (3.52) eşitsizlikleri elde edilir. Bu eşitsizlikler yardımıyla b_1 ve b_2 parametrelerine göre bu devrenin gerçekleyebildiği kutupların z domenindeki yerleşimi Şekil 3.15'deki gibi çizilebilir.

$$0 < b_2 < 1 \quad (3.52a)$$

$$b_1 > -2 \quad (3.52b)$$

$$b_1 + b_2 > -1 \quad (3.52c)$$



Şekil 3.15 CCII-SC devresi için gerçekleştirilebilir kutup yerleşim bölgesi (Taralı alan)

Şekil 3.15'de görülen taralı alan, Şekil 3.14'deki kararlılık üçgeni ile karşılaştırılırsa CCII-SC aktif filtre devre topolojisinin kararlı kompleks ve aynı işaretli reel kutupları gerçekleyebildiği görülür. Ancak bu devre topolojisi ile b_2 'nin negatif kaldığı bölgedeki kutuplar bir başka deyişle ters işaretli reel kutuplar gerçekleştirilemez. Bunun dışında bu devreyi kararsız kılan fonksiyonların kutupları reel ve aynı işaretlidir.

(3.50) eşitliklerinde görülen kapasite değerleri normalize olmayan değerlerdir. Şekil 3.13(b)'de verilen genel CCII-SC aktif filtre yapısındaki, ilk gerilim toplamı devresinin çıkışının tamamen belirlenmesi devrenin istenildiği şekilde çalışması için

zorunludur. Eğer bu gerilim çok büyük olursa devredeki ikinci katı aşırı yükler ve lineer olmayan çalışma bölgesine sürükler. Eğer bu gerilim çok küçük olursa bu durumda da gürültüden kaynaklanan hatalar etkili olur.

Bu nedenle V ile ifade edilen çıkışın tamamen belirlenmesi gerekmektedir. Bunun yapılabilmesi için T' fonksiyonuna normalizasyon işlemi uygulanır. Bu işlem yapılırken T fonksiyonunun değişmemesi gerektiğinden sadece C ve D kapasiteleri normalize edilir.

$$T' \rightarrow \mu T' \quad (3.53a)$$

$$(C, D) \rightarrow \left(\frac{C}{\mu}, \frac{D}{\mu} \right) \quad (3.53b)$$

İkinci normalizasyon işlemi ise devredeki minimum kapasite 5-10pf değerinde olması için yapılır. Minimum kapasitenin bu düzeyde belirlenmesinin nedeni parazitik kapasitelerin etkisini azaltmaktır. Böylece hem minimum kapasite hem de istenen maksimum kapasite oranlarının belirlenmesi mümkün olur. Devrede yer alan iki kapasite grubu vardır ki birbirlerinden bağımsız olarak normalize edilebilirler. Bunlar (A, B, C) ve (D, E, F, G, H) olarak belirlenirler. Devre incelendiğinde birinci grubun devredeki ilk toplama devresine ait olduğu, ikinci grubunda ikinci toplama devresine ait olduğu anlaşılmaktadır.

Buraya kadar genel ikinci dereceden aktif filtre transfer fonksiyonu, bu fonksiyonun işaret akış diyagramı, bu diyagramdan devreye nasıl geçildiği ve elemanların nasıl belirlendiği üzerinde durulmuştur. Şimdi bu yapıları özelleştirerek alçak geçiren, yüksek geçiren, band geçiren ve notch tipi filtre yapılarının nasıl elde edileceği incelenecektir.

3.5.2 İkinci Dereceden Alçak Geçiren Filtre Yapısı

Bilineer dönüşüm ile elde edilen ikinci dereceden z domeni alçak geçiren filtre transfer fonksiyonu (3.54)' de görülmektedir. Bu fonksiyona ilişkin katsayılar (3.34)

$$T(z) = \frac{K(1+z^{-1})^2}{1+b_1z^{-1}+b_2z^{-2}} \quad (3.54)$$

eşitliğinde görülen ikinci dereceden genel filtre transfer fonksiyonun katsayıları ile eşitlenirse (3.55)'de görülen eşitlikler yazılabilir.

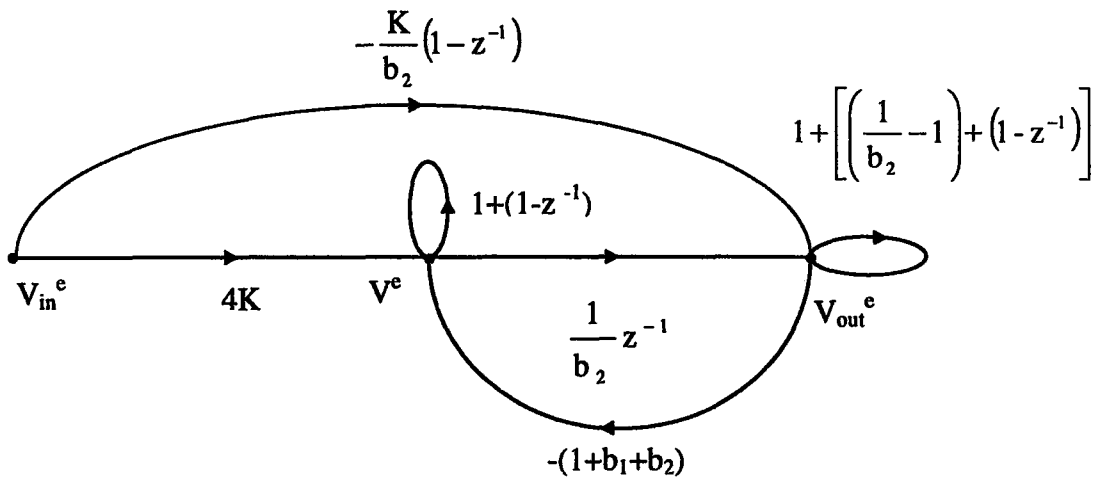
$$a_0 = K \quad (3.55a)$$

$$a_1 = 2K \quad (3.55b)$$

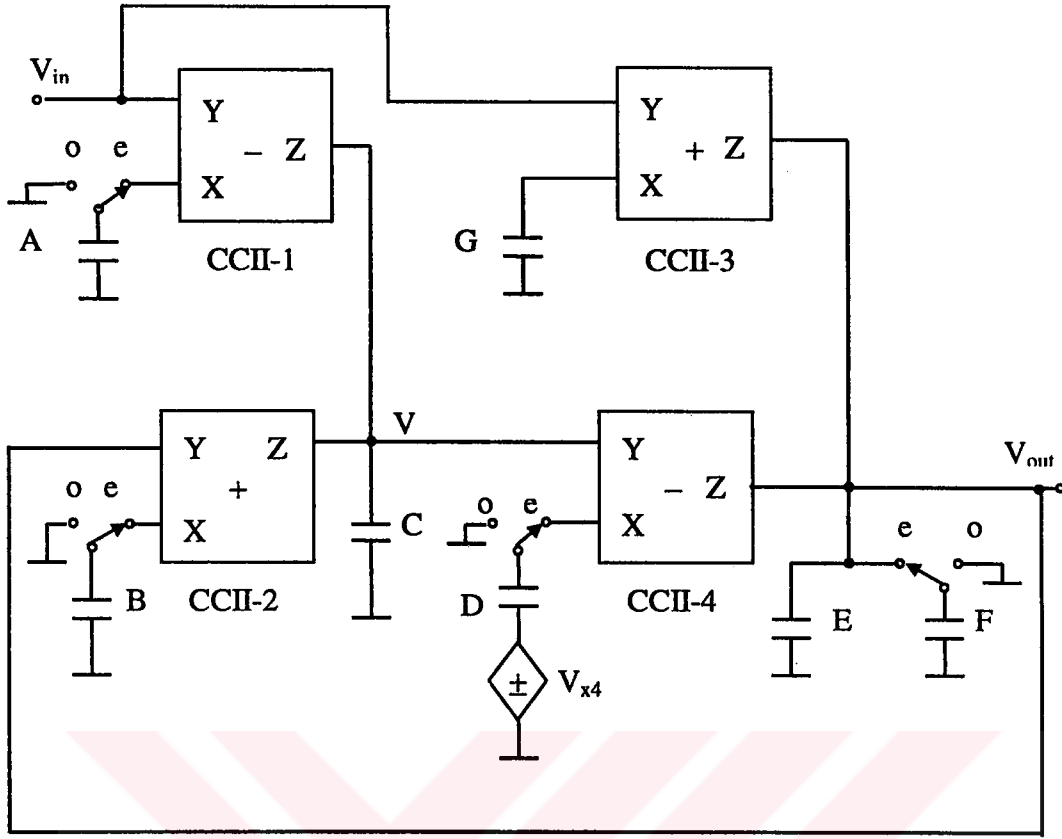
$$a_2 = K \quad (3.55c)$$

Bu eşitlikler dikkate alınarak Şekil 3.12(a)'daki işaret akış diyagramı yeniden oluşturulsa Şekil 3.16'da görülen işaret akış diyagramı elde edilir.

Bu işaret akış diyagramını dikkate alınarak CCII-SC aktif filtrelerin sentezi yöntemi uyarınca, işaret akış diyagramının negatif kazançlı dalları pozitif kazançlı akım taşıyıcıya, pozitif kazançlı dalları negatif kazançlı akım taşıyıcıya bağlanırken, dal kazançlarının mutlak değerleri de ilgili akım taşıyıcılara bağlanan kapasitelerin değerlerini verecek şekilde devre oluşturulursa Şekil 3.17'de görülen ikinci derece alçak geçiren aktif filtre yapısı elde edilir.



Şekil 3.16 İkinci dereceden alçak geçiren aktif filtre fonksiyonun işaret akış diyagramı



Şekil 3.17 İkinci dereceden alçak geçiren aktif filtre yapısı

Alçak geçiren filtre yapısı, genel filtre yapısı ile kıyaslandığında bu yapıda G ve H kapasite çifti ile bunların bağlı oldukları akım taşıyıcılar yerine tek bir kapasite ve tek bir akım taşıyıcı kullanıldığı görülür. Bunun nedeni a_0 ve a_2 katsayılarının birbirine eşit olmasıdır. Şekil 3.12(a) ve Şekil 3.16'da verilen işaret akış diyagramları incelendiğinde bunun yapılabileceği görülür. Devre elemanlarının, devre fonksiyonunun katsayıları cinsinden ifadeleri (3.56) eşitliklerinde verilmiştir. Bu yapının T ve T' transfer fonksiyonları (3.57) eşitliklerinde görülmektedir.

$$A = a_0 + a_1 + a_2 = 4K \quad (3.56a)$$

$$B = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.56b)$$

$$C = 1 \quad (3.56c)$$

$$D = \frac{1}{b_2} \quad (3.56d)$$

$$E = 1 \quad (3.56e)$$

$$F = \frac{1}{b_2} - 1 \quad (3.56f)$$

$$G = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.56g)$$

$$H = \frac{a_2}{b_2} = \frac{K}{b_2} \quad (3.56h)$$

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{GC + (DA - 2GC)z^{-1} + GCz^{-2}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.57a)$$

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{(BG - AE - AF) + (AE - BG)z^{-1}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.57b)$$

Buraya kadar yapılan incelemede K değeri pozitif olarak kabul edilmiştir. K değerinin negatif olması durumunda, kapasite değerleri dal kazançlarının mutlak değerleri tarafından belirlendiğinden yine pozitif olur. Ancak transfer fonksiyonunun değişmemesi için birinci ve üçüncü akım taşıyıcılar, akım kazançları -1 ile çarpılarak devreye konmalıdır. Böylece eviren tip devre de gerçekleşmiş olur.

3.5.3 İkinci Dereceden Yüksek Geçiren Filtre Yapısı

Bilineer dönüşüm ile elde edilen ikinci dereceden z domeni yüksek geçiren filtre transfer fonksiyonu (3.58)'de görülmektedir. Bu fonksiyona ilişkin katsayıları (3.34) eşitliğinde görülen ikinci dereceden genel filtre transfer fonksiyonun katsayıları ile eşitlenirse (3.59)'da görülen eşitlikler yazılabilir.

$$T(z) = \frac{K(1-z^{-1})^2}{1+b_1z^{-1}+b_2z^{-2}} \quad (3.58)$$

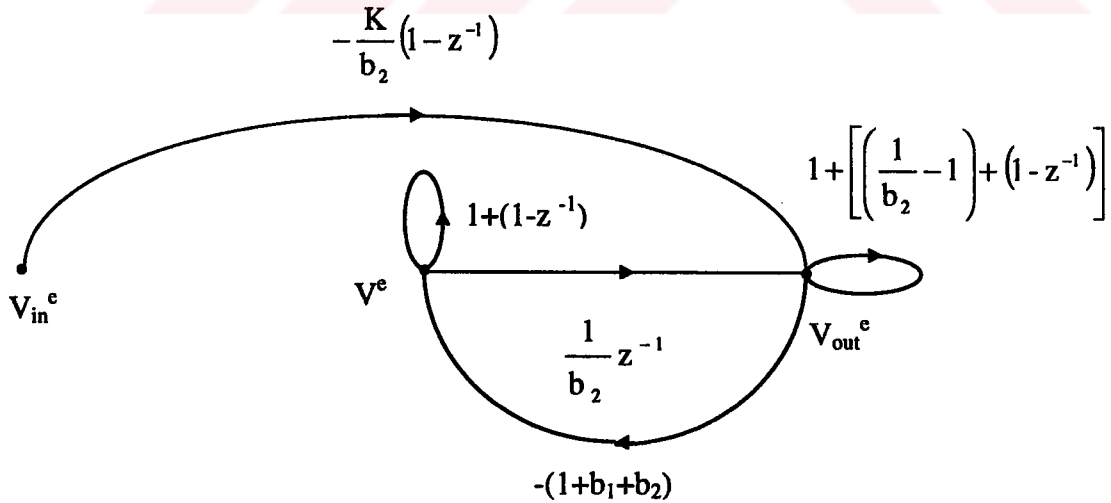
$$a_0 = K \quad (3.59a)$$

$$a_1 = -2K \quad (3.59b)$$

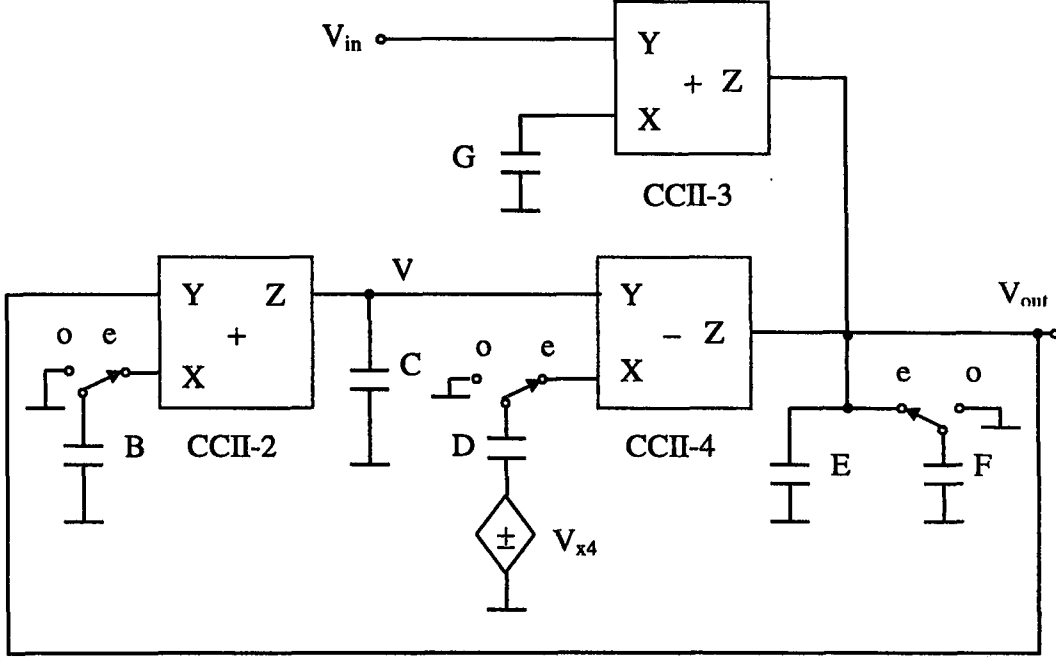
$$a_2 = K \quad (3.59c)$$

Bu eşitlikler dikkate alınarak Şekil 3.12(a)'daki işaret akış diyagramı yeniden oluşturulsa Şekil 3.18'de görülen işaret akış diyagramı elde edilir.

Bu işaret akış diyagramı dikkate alınarak CCII-SC aktif filtrelerin sentezi yöntemi uyarınca, işaret akış diyagramının negatif kazançlı dalları pozitif kazançlı akım taşıyıcıya, pozitif kazançlı dalları negatif kazançlı akım taşıyıcıya bağlanırken, dal kazançlarının mutlak değerleri de ilgili akım taşıyıcılara bağlanan kapasitelerin değerlerini verecek şekilde devre oluşturulursa Şekil 3.19'da görülen ikinci dereceden yüksek geçiren aktif filtre yapısı elde edilir.



Şekil 3.18 İkinci dereceden yüksek geçiren aktif filtre fonksiyonunun işaret akış diyagramı



Şekil 3.19 İkinci dereceden yüksek geçiren filtre yapısı

Yüksek geçiren filtre yapısı, genel filtre yapısı ile kıyaslandığında bu yapıda G ve H kapasite çifti ile bunların bağlı oldukları akım taşıyıcılar yerine tek bir kapasite ve tek bir akım taşıyıcı kullanıldığı görülür. Bunun nedeni a_0 ve a_2 katsayılarının birbirine eşit olmasıdır. Ayrıca A kapasitesi değeri, a_0 , a_1 , a_2 toplamı sıfır olduğundan sıfırdır. Bu nedenle A kapasitesi ve bağlı olduğu akım taşıyıcı devreden düşmüştür. Şekil 3.12(a) ve Şekil 3.18'de verilen işaret akış diyagramları incelendiğinde bunların yapılabileceği görülür. Devre elemanlarının devre fonksiyonunun katsayıları cinsinden ifadeleri (3.60) eşitliklerinde verilmiştir. Bu yapının T ve T' transfer fonksiyonları sırasıyla (3.61a) ve (3.62b) eşitliklerinde görülmektedir.

$$A = a_0 + a_1 + a_2 = 0 \quad (3.60a)$$

$$B = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.60b)$$

$$C = 1 \quad (3.60c)$$

$$D = \frac{1}{b_2} \quad (3.60d)$$

$$E = 1 \quad (3.60e)$$

$$F = \frac{1}{b_2} - 1 \quad (3.60f)$$

$$G = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.60g)$$

$$H = \frac{a_2}{b_2} = \frac{K}{b_2} \quad (3.60h)$$

Buraya kadar yapılan incelemede K değeri pozitif kabul edilmiştir. K değerinin

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{GC + (-2GC)z^{-1} + GCz^{-2}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.61a)$$

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{BG + (-BG)z^{-1}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.61b)$$

negatif olması durumunda, kapasite değerleri dal kazançlarının mutlak değerleri tarafından belirlendiğinden yine pozitif olur. Ancak transfer fonksiyonunun değişmemesi için üçüncü akım taşıyıcı, akım kazancı -1 ile çarpılarak devreye konmalıdır. Böylece eviren tip devre de gerçekleşmiş olur.

3.5.4 İkinci Dereceden Band Geçiren Filtre Yapısı

Bilineer dönüşüm ile elde edilen ikinci dereceden z domeni band geçiren filtre fonksiyonu (3.62) eşitliğinde görülmektedir. Bu fonksiyona ilişkin katsayılar, (3.34)

$$T(z) = \frac{K(1 - z^{-1})(1 + z^{-1})}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (3.62)$$

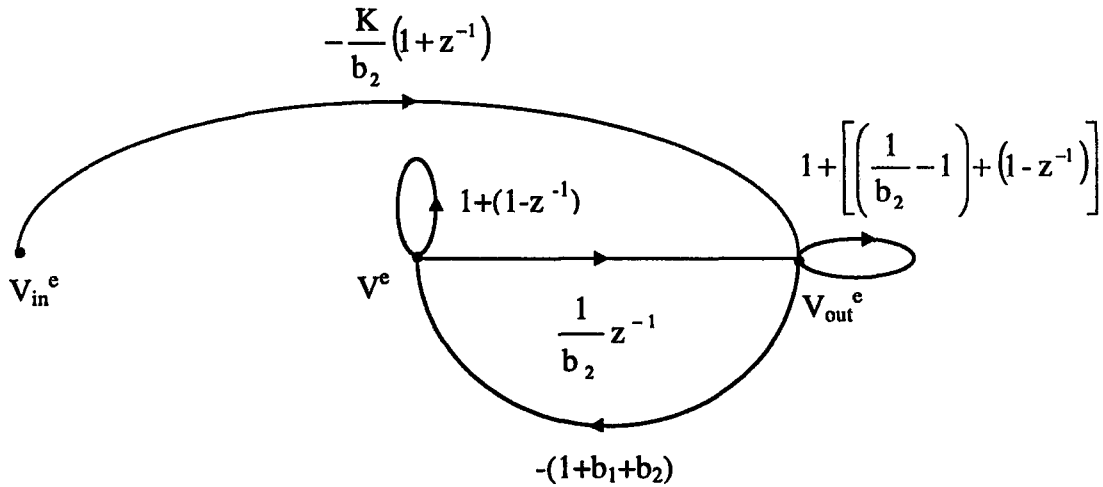
eşitliğinde görülen ikinci dereceden genel filtre transfer fonksiyonun katsayıları ile eşitlenirse (3.63)'de görülen eşitlikler yazılabilir.

$$a_0 = K \quad (3.63a)$$

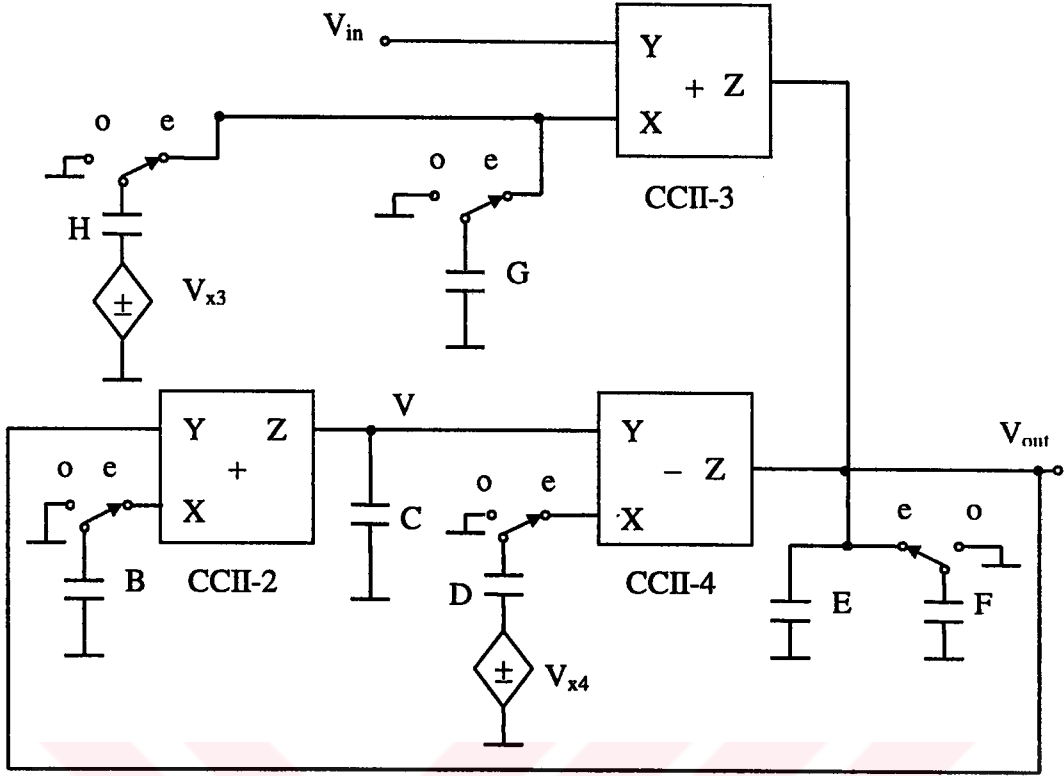
$$a_1 = 0 \quad (3.63b)$$

$$a_2 = -K \quad (3.63c)$$

Bu eşitlikler dikkate alınarak Şekil 3.12(a)'daki işaret akış diyagramı yeniden oluşturulsa Şekil 3.20'de görülen işaret akış diyagramı elde edilir. Bu işaret akış diyagramı dikkate alınarak CCII-SC aktif filtrelerin sentezi yöntemi uyarınca, işaret akış diyagramının negatif kazançlı dalları pozitif kazançlı akım taşıyıcıya, pozitif kazançlı dalları negatif kazançlı akım taşıyıcıya bağlanırken, dal kazançlarının mutlak değerleri de ilgili akım taşıyıcılara bağlanan kapasitelerin değerlerini verecek şekilde devre oluşturulursa Şekil 3.21'de görülen ikinci derece band geçiren aktif filtre yapısı elde edilir.



Şekil 3.20 İkinci dereceden band geçiren aktif filtre fonksiyonun işaret akış diyagramı



Şekil 3.21 İkinci derece band geçiren aktif filtre yapısı

Band geçiren filtre yapısı, genel filtre yapısı ile kıyaslandığında bu yapıda A kapasitesinin değerin sıfır olduğu görülür. Çünkü a_0 , a_1 , a_2 katsayılarının toplamı sıfırdır. Bu nedenden ötürü A kapasitesi ve bağlı olduğu akım taşıyıcı devre dışı kalmıştır. Bu yapıda G ve H kapasite çiftinin, genel yapıdan farklı olarak birbirlerine paralel ve tek bir pozitif kazançlı akım taşıyıcıya bağlandığı görülür. Bunun nedeni a_0 ve a_2 katsayılarının zıt işaretli ve birbirine eşit olmasıdır. Devre elemanlarının devre fonksiyonunun katsayıları cinsinden ifadeleri (3.64) eşitliklerinde verilmiştir. Bu yapının T ve T' transfer fonksiyonları sırasıyla (3.65a) ve (3.65b) eşitliklerinde görülmektedir.

$$A = a_0 + a_1 + a_2 = 0 \quad (3.64a)$$

$$B = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.64b)$$

$$C = 1 \quad (3.64c)$$

$$D = \frac{1}{b_2} \quad (3.64d)$$

$$E = 1 \quad (3.64e)$$

$$F = \frac{1}{b_2} - 1 \quad (3.64f)$$

$$G = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.64g)$$

$$H = \left| \frac{a_2}{b_2} \right| = \frac{K}{b_2} \quad (3.64h)$$

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{GC - GCz^{-2}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.65a)$$

$$T = \frac{V_{out}^e}{V_{in}^e} = \frac{BG - BGz^{-1}}{(EC + FC) + (DB - 2EC - FC)z^{-1} + ECz^{-2}} \quad (3.65b)$$

Buraya kadar yapılan incelemede K değeri pozitif kabul edilmiştir. K değerinin negatif olması durumunda, kapasite değerleri dal kazançlarının mutlak değerleri tarafından belirlendiğinden yine pozitif olur. Ancak transfer fonksiyonunun değişmemesi için üçüncü akım taşıyıcı, akım kazancı -1 ile çarpılarak devreye konmalıdır. Böylece eviren tip devre de gerçekleşmiş olur.

3.5.5 İkinci Dereceden Notch Filtre Yapıları

Bilineer dönüşüm ile elde edilen ikinci dereceden z domenli notch tipi filtre fonksiyonu (3.66)'da görülmektedir.

$$T(z) = \frac{K(1 + \varepsilon z^{-1} + z^{-2})}{1 + b_1 z^{-1} + b_2 z^{-2}} \quad (3.66)$$

Transfer fonksiyonunda görülen ε değerinin, b_1 ve b_2 katsayıları ile arasındaki ilişkiye göre (3.66)'da görülen transfer fonksiyonu alçak geçiren notch, yüksek geçiren notch veya band söndüren filtre tiplerini gerçekler. Eğer ε , $b_1/\sqrt{b_2}$ 'den büyükse ω_p , ω_z 'den küçük demektir ve transfer fonksiyonu alçak geçiren notch tipi filtreyi gerçekler. Eğer ε katsayısı, $b_1/\sqrt{b_2}$ 'den küçükse ω_p , ω_z 'den büyük demektir ve transfer fonksiyonu yüksek geçiren notch tipi filtreyi gerçekler. Eğer ε , $b_1/\sqrt{b_2}$ 'ye eşitse ω_p , ω_z 'ye eşit demektir ve transfer fonksiyonu band söndüren filtre tipini gerçekler. Her üç filtre tipini gerçekleyen devrenin topolojisi, tek bir fark haricinde alçak geçiren filtre devre topolojisinin aynısıdır. Bu fark A kapasitesinin değerinden kaynaklanmaktadır. Notch tipi filtrelere ilişkin kapasite değerlerinin (3.66)'deki fonksiyonun katsayıları cinsinden ifadeleri (3.67) eşitliklerinde görülmektedir.

$$A = a_0 + a_1 + a_2 = K(2 + \varepsilon) \quad (3.67a)$$

$$B = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.67b)$$

$$C = 1 \quad (3.67c)$$

$$D = \frac{1}{b_2} \quad (3.67d)$$

$$E = 1 \quad (3.67e)$$

$$F = \frac{1}{b_2} - 1 \quad (3.67f)$$

$$G = \frac{a_0}{b_2} = \frac{K}{b_2} \quad (3.67g)$$

$$H = \frac{a_2}{b_2} = \frac{K}{b_2} \quad (3.67h)$$

K katsayısı buraya kadar olan incelemede pozitif olarak kabul edilmiştir. Eğer K katsayısı negatif olursa (3.67) eşitliklerinde görülen K katsayısının yerini mutlak değer K alır ve böylece kapasite değerlerinin negatif değer alması önlenir. K'nın negatif olmasından dolayı devrede yer alan bir ve üç numaralı akım taşıyıcıların akım kazançları -1 ile çarpılmalıdır. Böylece notch filtrelerin eviren tipleri de gerçekleşmiş olur.

3.6 Duyarlılık Analizi

Tasarımı yapılan bir devreye ilişkin devre parametreleri, gerçekleştirilmesindeki amaca uygun olarak analiz veya sentez yöntemleri ile belirlenir. Belirlenen bu değerlerin değişmemesi istenir. Ancak sıcaklık, nem, yaşlanma, gibi bazı çevre koşullarının elemanların üretilmesinde kullanılan malzemeleri etkilemesi, devre elemanlarının üretilmesinde oluşacak üretim yanlıgıları devre parametrelerinde engellenemez bir takım değişikliklere neden olur. Bir başka deyişle meydana gelen bu değişimler transfer fonksiyonunda, transfer fonksiyonun genliğinde, fazında, sıfır ve kutuplarında, pay ve payda polinomlarının katsayılarında değişimlere yol açar. Diğer taraftan tasarlanan devrenin beklenen davranışı göstermesi için bu değişimlerin belirli sınırlar içerisinde kalması gerekir.

Herhangi bir devre parametresinde oluşacak küçük değişimlerin, devreyi belirleyen büyüklükte oluşturacağı değişimleri bulmak amacıyla bir-parametrelili duyarlılık fonksiyonu tanımlanmıştır. Herhangi bir devre parametresi x ile, devreyi belirleyen büyüklük de $y = y(x)$ ile gösterilsin. x 'te meydana gelecek küçük bir değişime karşılık y 'de meydana gelecek değişimi bulmak amacıyla tanımlanan ve $S(y, x)$ ile gösterilen bir parametrelili duyarlılık fonksiyonu dört ayrı biçimde tanımlanır ve adlandırılır.

1. Normalleştirilmiş duyarlılık :

$$\frac{\Delta y}{y} = S(y, x) \frac{\Delta x}{x} \quad , \quad S(y, x) = \frac{\partial \ln y}{\partial \ln x} \quad (3.68a)$$

2. y'ye göre yarı normalleştirilmiş duyarlılık :

$$\Delta y = S(y, x) \frac{\Delta x}{x} \quad , \quad S(y, x) = \frac{\partial y}{\partial \ln x} \quad (3.68b)$$

3. x'e göre yarı normalleştirilmiş duyarlılık :

$$\frac{\Delta y}{y} = S(y, x) \Delta x \quad , \quad S(y, x) = \frac{\partial \ln y}{\partial x} \quad (3.68c)$$

4. Normalleştirilmemiş duyarlılık :

$$\Delta y = S(y, x) \Delta x \quad , \quad S(y, x) = \frac{\partial y}{\partial x} \quad (3.68d)$$

y'deki değişmeyi gösteren Δy de aşağıdaki biçimde tanımlanmıştır :

$$\Delta y = y(x + \Delta x) - y(x) \quad (3.68e)$$

Devreyi belirleyen büyüklüğün ve devre parametresinin türüne bağlı olarak bu tanımlardan biri kullanılır. Bu eşitlikler Δy 'yi x civarında Taylor serisine açarak ve birinci terim yanında diğerlerini ihmal ederek bulunur. Bunun için de x 'teki değişimin küçük olması gerekir.

Birden çok devre parametresinde aynı anda küçük değişimlerin oluşması durumunda devreyi belirleyen büyüklükte oluşacak değişimin belirlenebilmesi amacıyla da çok-parametrelili duyarlılık tanımlanmıştır. Devrede k tane devre parametresi olduğunu kabul edelim. i 'inci devre parametresi x_i , $i=1,2,...,k$ ile, x_i 'deki

değişme Δx_i ile ve x_i 'deki normalleştirilmiş değişme $\Delta x_i/x_i$ ile gösterilsin. X sütun vektörü, X 'teki değişmeyi gösteren ΔX vektörü, X 'teki normalleştirilmiş değişmeyi gösteren $\Delta \hat{X}$ vektörü ve y 'ye ilişkin çok-parametrelili duyarlılık vektörü (3.69)'da tanımlanmıştır.

$$X = [x_1 \quad x_2 \quad . \quad . \quad x_k]^t \quad (3.69a)$$

$$\Delta X = [\Delta x_1 \quad \Delta x_2 \quad . \quad . \quad \Delta x_k]^t \quad (3.69b)$$

$$\Delta \hat{X} = \left[\frac{\Delta x_1}{x_1} \quad \frac{\Delta x_2}{x_2} \quad . \quad . \quad \frac{\Delta x_k}{x_k} \right]^t \quad (3.69c)$$

$$S(y, x) = [S(y, x_1) \quad S(y, x_2) \quad . \quad . \quad S(y, x_k)]^t \quad (3.69d)$$

O halde y 'ye ilişkin normalleştirilmiş çok parametrelili duyarlılık fonksiyonu (3.70)'deki gibi tanılanır.

$$\frac{\Delta y}{y} = S'(y, X) \Delta \hat{X} \quad (3.70)$$

Diğer tür tek-parametrelili duyarlılık tanımları için de benzer yolla çok-parametrelili duyarlılık tanımları elde edilebilir.

3.6.1 Genlik Fonksiyonu Duyarlılığı

Devrede yer alan kapasitelerde meydana gelebilecek küçük değişimlerin genlik fonksiyonunda oluşturacağı değişimi tespit etmek için genlik fonksiyonu duyarlılığı incelenmelidir. Akım taşıyıcı içeren SC aktif filtre devre topolojisinin gerçeklediği transfer fonksiyonu (3.48) eşitliğinde verilmişti. Bu eşitlikte z yerine $e^{j\omega T}$ konur ve gerekli işlemler yapılırsa transfer fonksiyonuna ilişkin genlik fonksiyonu :

$$k_1 = GC + (DA - GC - HC) \cdot \cos(\omega T) + HC \cdot \cos(2\omega T) \quad (3.71a)$$

$$k_2 = (DA - GC - HC) \cdot \sin(\omega T) + HC \cdot \sin(2\omega T) \quad (3.71b)$$

$$k_3 = C(E + F) + (DB - FC - 2EC) \cdot \cos(\omega T) + EC \cdot \cos(2\omega T) \quad (3.71c)$$

$$k_4 = (DB - FC - 2EC) \cdot \sin(\omega T) + EC \cdot \sin(2\omega T) \quad (3.71d)$$

olmak üzere;

$$|T| = \sqrt{\frac{k_1^2 + k_2^2}{k_3^2 + k_4^2}} \quad (3.72)$$

olarak bulunur. Genlik fonksiyonu, devre parametrelerine göre duyarlılığı (3.73) eşitliğinde tanımlanmıştır.

$$S_x^{[T]} = \frac{x}{|T|} \frac{\partial |T|}{\partial x} \quad (3.73)$$

Bu eşitlik uyarınca genlik fonksiyonunun A, B, C, D, E, F, G, H kapasitelerine göre duyarlılıkları hesaplanırsa (3.74) eşitliklerinde görülen duyarlılık fonksiyonları elde edilir.

$$S_A^{[T]} = \frac{AD}{k_1^2 + k_2^2} \cdot (k_1 \cos(\omega T) + k_2 \sin(\omega T)) \quad (3.74a)$$

$$S_B^{[T]} = -\frac{BD}{k_3^2 + k_4^2} \cdot (k_3 \cos(\omega T) + k_4 \sin(\omega T)) \quad (3.74b)$$

$$S_C^{[T]} = \frac{C}{k_1^2 + k_2^2} \cdot \left(k_1 (G - (G + H)\cos(\omega T) + H\cos(2\omega T)) + \right. \\ \left. - \frac{C}{k_3^2 + k_4^2} \cdot \left(k_3 (E + F - (F + 2E)\cos(\omega T) + E\cos(2\omega T)) + \right. \right. \\ \left. \left. - \frac{C}{k_3^2 + k_4^2} \cdot \left(k_4 ((-F - 2E)\sin(\omega T) + E\sin(2\omega T)) \right) \right) \right) \quad (3.74c)$$

$$S_D^{[T]} = S_A^{[T]} + S_B^{[T]} \quad (3.74d)$$

$$S_E^{[T]} = -\frac{EC}{k_3^2 + k_4^2} (k_3 (1 - 2\cos(\omega T) + \cos(2\omega T)) + k_4 (-2\sin(\omega T) + \sin(2\omega T))) \quad (3.74e)$$

$$S_F^{[T]} = -\frac{FC}{k_3^2 + k_4^2} (k_3 (1 - \cos(\omega T)) + k_4 (-\sin(\omega T))) \quad (3.74f)$$

$$S_G^{[T]} = \frac{GC}{k_1^2 + k_2^2} (k_1 (1 - \cos(\omega T)) + k_2 (-\sin(\omega T))) \quad (3.74g)$$

$$S_H^{[T]} = \frac{HC}{k_1^2 + k_2^2} (k_1 (-\cos(\omega T) + \cos(2\omega T)) + k_2 (-\sin(\omega T) + \sin(2\omega T))) \quad (3.74h)$$

Bu şekilde bulunan duyarlılık fonksiyonları (3.68a) eşitliğinde yerine konursa herbir devre parametresinin $|T|$ fonksiyonunda oluşturacağı normalleştirilmiş değişme ayrı ayrı bulunabilir. Genlik fonksiyonuna ilişkin çok parametrelili duyarlılık vektörü de (3.75) eşitliğinde görülmektedir.

$$S(|T|, X) = [S_A^{[T]} \quad S_B^{[T]} \quad S_C^{[T]} \quad S_D^{[T]} \quad S_E^{[T]} \quad S_F^{[T]} \quad S_G^{[T]} \quad S_H^{[T]}]^T \quad (3.75)$$

O halde tüm devre parametrelerinde aynı anda meydana gelebilecek küçük değişimlerin genlik fonksiyonunda oluşturacağı normalleştirilmiş değişim, (3.69) ve (3.70) eşitlikleri dikkate alınarak (3.76) eşitliğinde görüldüğü gibi elde edilir.

$$S(T, X) = \begin{bmatrix} S_A^{[T]} & S_B^{[T]} & S_C^{[T]} & S_D^{[T]} & S_E^{[T]} & S_F^{[T]} & S_G^{[T]} & S_H^{[T]} \end{bmatrix} \cdot \begin{bmatrix} \Delta A/A \\ \Delta B/B \\ \Delta C/C \\ \Delta D/D \\ \Delta E/E \\ \Delta F/F \\ \Delta G/G \\ \Delta H/H \end{bmatrix} \quad (3.76)$$

Kesim frekansı 2kHz, söndürme frekansı 6kHz, örnekleme frekansı 36kHz ($T = 27.7778\mu s$), kazancı 0dB olan bir alçak geçiren filtre için gerekli işlemler yapılırsa, $A = 4pf$, $B = 4pf$, $C = 16.5972pf$, $D = 66.0552pf$, $E = 100.755pf$, $F = 64.38pf$, $G = 4pf$ olarak bulunur. Bu kapasiteler için duyarlılık fonksiyonları kesim frekansı olan 2kHz'de hesaplanırsa, $S_A^{[T]} = 1.031253$, $S_B^{[T]} = 0.003793$, $S_C^{[T]} = -1.035046$, $S_D^{[T]} = 1.035046$, $S_E^{[T]} = -2.896 \cdot 10^{-3}$, $S_F^{[T]} = -1.000898$, $S_G^{[T]} = -0.031253$ olarak bulunur. Eğer tüm kapasiteler için bağıl değişimin %1 olduğu kabul edilirse genlik fonksiyonuna ilişkin normalleştirilmiş değişim (3.76)'da verilen eşitlik yardımıyla $\Delta|T|/|T| \cong \%0$ olarak bulunur. Genlik fonksiyonun her bir kapasite için frekansla değişimi Şekil 3.22'de verilmiştir.

3.6.2 Katsayı Duyarlılığı

Genel ikinci dereceden z domeni filtre transfer fonksiyonu (3.34) eşitliğinde verilmişti. Bu eşitliğin paydasında yer alan b_1 , b_2 katsayılarının belirli sınırlar içerisinde kalması gerekmektedir. Bu sınırlar (3.35) eşitliğinde verilmişti. Bu katsayıların sınır değerlere yakın olması durumunda, devre parametrelerinde meydana gelecek küçük değişimler bu katsayıları sınırların dışına çıkartabilir. Bu durumdan kutuplar doğrudan etkilenirler ve kararsız bölgeye geçebilirler. Bu nedenden ötürü katsayı duyarlılığının hesaplanarak böyle bir durumun söz konusu olup olmayacağının incelemesinde yarar vardır. Katsayı duyarlılığının hesabında normalleştirilmiş duyarlılık kullanılır.

Akım taşıyıcı içeren SC aktif filtrelerin transfer fonksiyonu daha önce (3.48) eşitliğinde verilmişti. Bu eşitlikten b_1 ve b_2 katsayıları :

$$b_1 = \frac{DB - FC - 2EC}{EC + FC} \quad (3.77a)$$

$$b_2 = \frac{E}{E + F} \quad (3.77b)$$

olarak bulunabilir. Bu katsayıların kapasitelere göre duyarlılıkları aşağıda görülen eşitliklerden elde edilir.

$$S(b_2, E) = \frac{F}{F + E} \quad (3.78a)$$

$$S(b_2, F) = -\frac{F}{F + E} \quad (3.78b)$$

$$S(b_1, D) = S(b_1, B) = \frac{BD}{BD - FC - 2EC} \quad (3.78c)$$

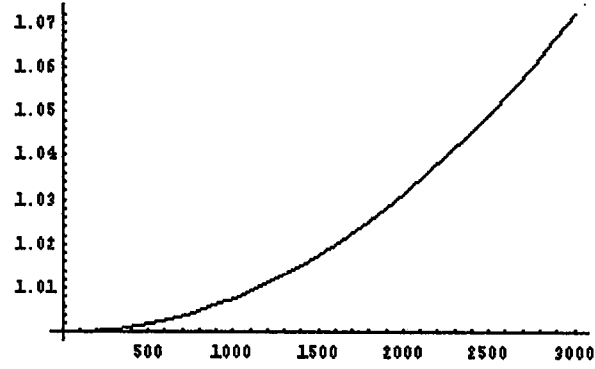
$$S(b_1, E) = -\frac{E}{E + F} \cdot \frac{FC + BD}{BD - FC - 2EC} \quad (3.78d)$$

$$S(b_1, F) = \frac{F}{E + F} \cdot \frac{EC - BD}{BD - FC - 2EC} \quad (3.78e)$$

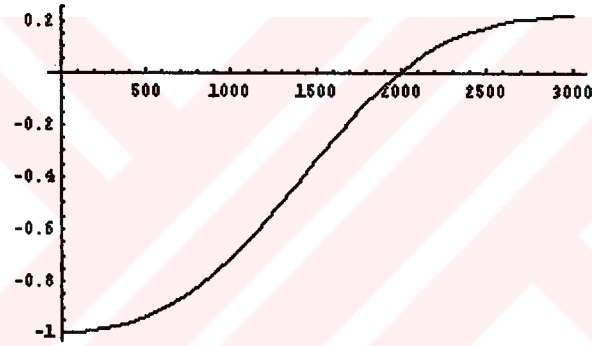
$$S(b_1, C) = -\frac{BD}{BD - FC - 2EC} \quad (3.78f)$$

Devrede yer alan tüm kapasitelerin bağıl değişimlerinin birbirine eşit olması durumunda b_1 ve b_2 katsayılarında meydana gelecek normalleştirilmiş bağıl değişimin sıfır olacağı (3.78) denklemlerinden açıkça görülmektedir. Bir başka deyişle kapasitelerin bağıl değişimleri eşit olduğu zaman kapasitelerde oluşan

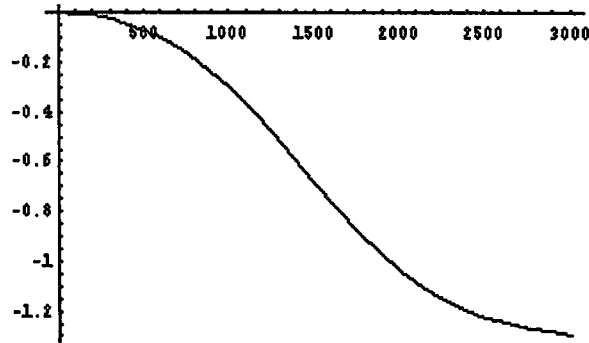
değişimler birbirini kompanse etmektedir. Bunun sonucunda da b_1 ve b_2 katsayıları ile bu katsayılar tarafından belirlenen kutuplarda bir değişim olmamaktadır.



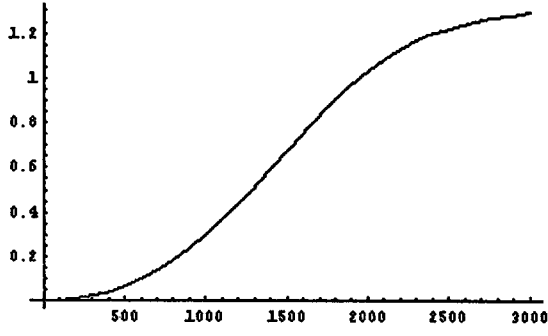
(a)



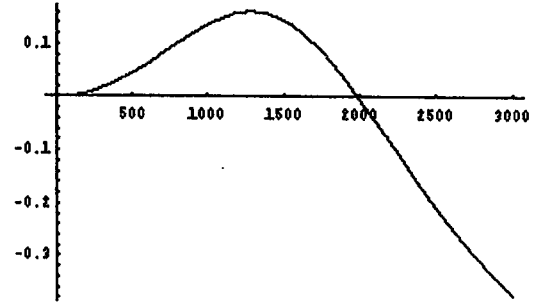
(b)



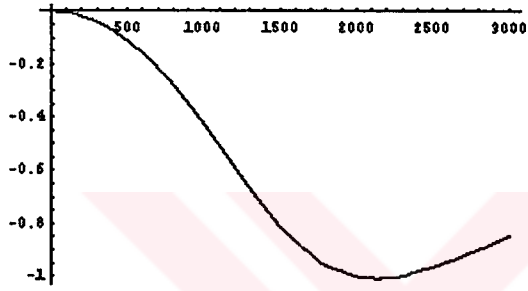
(c)



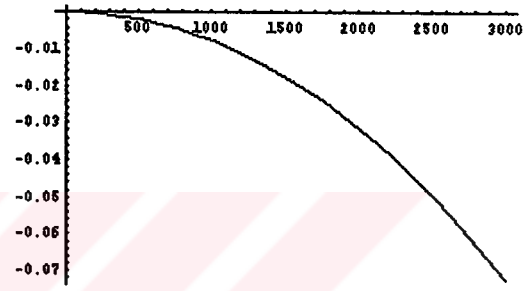
(d)



(e)



(f)



(g)

Şekil 3.22 Genlik transfer fonksiyonunun

(a) A kapasitesine

(b) B kapasitesine

(c) C kapasitesine

(d) D kapasitesine

(e) E kapasitesine

(f) F kapasitesine

(g) G kapasitesine göre duyarlılığının frekansla değişimi

BÖLÜM 4

AKIM TAŞIYICI-SC AKTİF FİLTRELER İLE İLGİLİ ÖRNEK TASARIMLAR

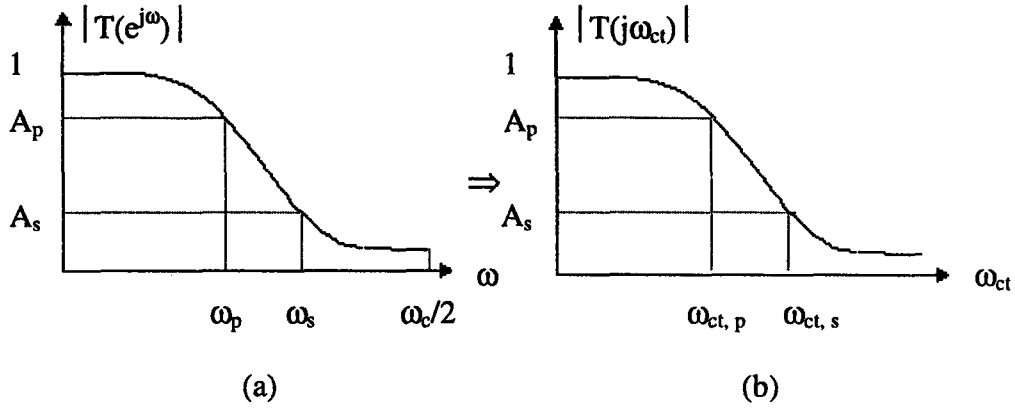
4.1 Bilineer Dönüşüm Metodu Kullanılarak Butterworth Filtre Tasarım Yöntemi

Z domeni transfer fonksiyonlarını elde etmek için s domeni transfer fonksiyonlarına bilinear dönüşüm uygulanır. Bu dönüşüm yapılırken dikkat edilmesi gereken en önemli nokta, $s=j\omega_{ct}$ biçiminde tanımlanan ω_{ct} analog frekansı ile z'nin $e^{j\omega\tau}$ 'ya eşit olduğu örneklenmiş ω veri domeni frekansı arasındaki lineer olmayan bağıntıdan ileri gelen frekans bozulmasıdır. Bu ilişki (4.1) eşitliğinde görülebilir. Bu

$$\omega_{ct} = \frac{2}{\tau} \tan\left(\frac{\omega\tau}{2}\right) \quad (4.1)$$

eşitlikte görülen τ örnekleme zamanıdır. Eğer $\omega\tau \ll 1$ şartı sağlanıyorsa bu etki önemini yitirir. Aksi durumlarda, örneklenmiş veri domeni frekanslarına çarpıtma işlemi uygulanarak bu frekanslar analog frekanslara çevrilmelidir. Elde edilen bu frekans karakteristikleri gözönüne alınarak s domeninde transfer fonksiyonu tasarımı yapılır. S domeni alçak geçiren butterworth filtre transfer fonksiyonunun tasarımı için öncelikle filtrenin derecesi bilinmelidir. Filtre derecesi N ile gösterilir ve (4.2)'de görülen eşitsizlik yardımı ile bulunur.

$$N \geq \frac{1}{2} \frac{\log\left[\left(\frac{1}{A_s^2} - 1\right) / \left(\frac{1}{A_p^2} - 1\right)\right]}{\log\left[\frac{\omega_{ct,s}}{\omega_{ct,p}}\right]} \quad (4.2)$$



Şekil 4.1 (a) Alçak geçiren filtrenin örneklenmiş veri domeni frekans karakteristikleri
(b) Alçak geçiren filtrenin analog domen frekans karakteristiği

$$\omega_{\text{cutoff}} = \frac{\omega_{\text{ct,p}}}{\left(\frac{1}{A_p^2} - 1 \right)^{1/2N}} \quad (4.3)$$

Derecesi bu şekilde belirlenen alçak geçiren transfer fonksiyonunun kesim frekansı ise (4.3) eşitliğinden elde edilir. Bu eşitlikde A_p , genliğin maksimum değerinin $1/\sqrt{2}$ 'sine eşitse kesim frekansı ω_p 'ye eşit olur. Kesim frekansı ve derecesi bu şekilde tespit edilen alçak geçiren filtre transfer fonksiyonu (4.4) eşitlikleri yardımıyla bulunur.

$$T(s) = \prod_{k=1}^{N/2} \frac{\omega_{\text{cutoff}}^2}{s^2 + \omega_{\text{cutoff}} b_k s + \omega_{\text{cutoff}}^2} \quad ; N \text{ çift} \quad (4.4a)$$

$$T(s) = \frac{\omega_{\text{cutoff}}}{s + \omega_{\text{cutoff}}} \prod_{k=1}^{(N/2)-1} \frac{\omega_{\text{cutoff}}^2}{s^2 + \omega_{\text{cutoff}} b_k s + \omega_{\text{cutoff}}^2} \quad ; N \text{ tek} \quad (4.4b)$$

$$b_k = 2 \sin \left[(2k-1) \frac{\pi}{2N} \right] \quad (4.4c)$$

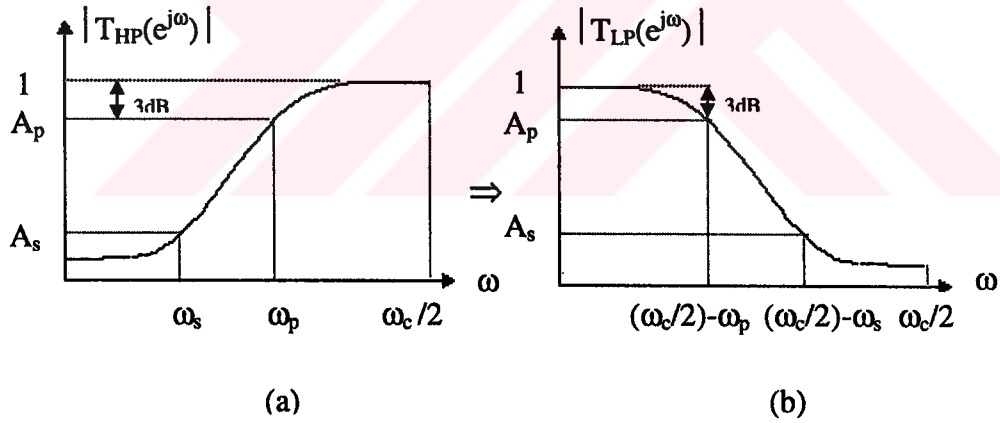
Elde edilen bu fonksiyonlara bilineer dönüşüm uygulanarak başlangıçta

belirlenen örneklenmiş veri domeni frekans karakteristiklerine sahip z domeni alçak geçiren transfer fonksiyonları elde edilebilir. Bunun için s yerine (4.5)'de görülen ifade konur.

$$s \rightarrow \frac{2}{\tau} \frac{1-z^{-1}}{1+z^{-1}} = 2f_c \frac{1-z^{-1}}{1+z^{-1}} \quad (4.5)$$

Eğer $\omega\tau \ll 1$ ise örneklenmiş veri domeni frekans karakteristiklerine çarpıtma işlemi uygulamaya gerek yoktur. Bu karakteristikler analog karakteristikler olarak kabul edilip buna göre z domeni transfer fonksiyonu elde edilir.

Yüksek geçiren bir filtrenin z domeni transfer fonksiyonunu elde etmek için aynalama yöntemi (mirror filter) kullanılabilir. Tasarlanmak istenen yüksek geçiren filtrenin örneklenmiş veri domeni frekans karakteristiği, alçak geçiren filtre karakteristiğine dönüştürülür. Örneklem frekansı ω_c olduğuna göre bu dönüşümün nasıl yapılacağı Şekil 4.2'de görülmektedir.



Şekil 4.2 (a) Örneklenmiş veri domeni yüksek geçiren filtrenin frekans karakteristiği
(b) Aynalama yöntemi ile elde edilen alçak geçiren eşdeğeri

Bu dönüşüm yapıldıktan sonra elde edilen örneklenmiş veri domeni alçak geçiren filtre karakteristiklerine göre z domeni alçak geçiren filtre transfer fonksiyonu tasarlanır. Bu fonksiyonda (4.6) eşitliğinde görülen dönüşüm yapılarak yani alçak geçiren filtre transfer fonksiyonunda z yerine $-z$ konarak z domeni yüksek geçiren filtre transfer fonksiyonu elde edilmiş olur.

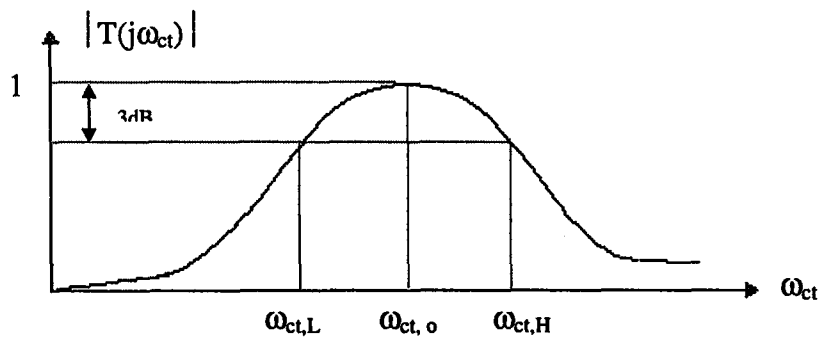
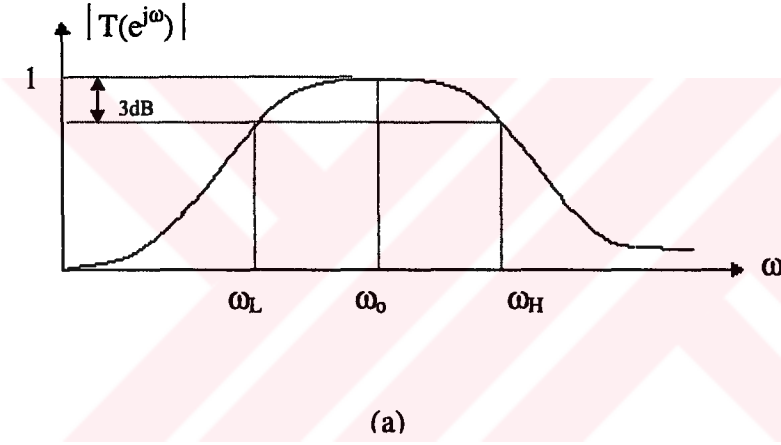
$$T_{HP}(z) = T_{LP}(-z) \quad (4.6)$$

Band geçiren filtrelerde ω_L alt kesim frekansı, ω_H üst kesim frekansı olmak üzere geçirme bandı merkez frekansı ve band genişliği :

$$\omega_o = \sqrt{\omega_L \omega_H} \quad (4.7)$$

$$B = \omega_H - \omega_L \quad (4.8)$$

eşitlikleri ile verilmektedir.



Şekil 4.3 (a) Band geçiren filtrenin örneklenmiş veri domeni frekans karakteristiği

(b) Band geçiren filtrenin analog domeni frekans karakteristiği

Bu eşitliklerde görülen alt ve üst kesim frekansları 3dB düşme noktalarıdır. Bu durumda ω_L ve ω_H için bulunan karmaşık ifadelerin taylor serisine açılmasıyla bulunan eşitlikler, $Q_p \geq 10$ şartı sağlandığı durumlarda :

$$\omega_L = \omega_o \left(1 - \frac{1}{Q_p} \right) \quad (4.9)$$

$$\omega_H = \omega_o \left(1 + \frac{1}{Q_p} \right) \quad (4.10)$$

$$B = \omega_H - \omega_L = \frac{\omega_o}{Q_p} \quad (4.11)$$

olarak elde edilir.

Örneklenmiş veri domeni frekans karakteristiği verilen, bir band geçiren filtrenin tasarlanması için öncelikle çarpıtma işlemi uygulanarak analog eşdeğeri elde edilmelidir. İkinci dereceden band geçiren filtrenin transfer fonksiyonu, transfer fonksiyonu (4.12)' de verilen birinci dereceden alçak geçiren filtrede s yerine (4.13)'de görülen :

$$T(s) = \frac{1}{1+s} \quad (4.12)$$

$$s \rightarrow \frac{s^2 + \omega_{ct,o}^2}{(\omega_{ct,H} - \omega_{ct,L})s} \quad (4.13)$$

ifade konarak elde edilir. Eğer dördüncü dereceden band geçiren filtre tasarlanacak ise o takdirde aynı dönüşüm ikinci derece alçak geçiren filtre transfer fonksiyonunda yerine konur. Daha üst dereceleri elde etmek için de benzer işlemler yapılır. Elde edilen bu çarpıtılmış s domeni transfer fonksiyonuna bilineer dönüşüm uygulanarak istenen transfer fonksiyonu elde edilir.

4.2 Akım Taşıyıcı-SC Aktif Filtrelerle İlgili Örnek Tasarımlar

Bu bölümde ikinci, üçüncü ve dördüncü dereceden alçak, yüksek ve band geçiren filtre örnekleri sunulmuştur.

4.2.1 Alçak Geçiren Filtre Devresi

Bir alçak geçiren filtreye ait örneklenmiş veri domeni karakteristikleri (4.14) eşitliklerinde görülmektedir. Bu filtrenin örnekleme zamanı $f_c=36\text{kHz}$ olarak seçilmiştir. Filtrenin $\omega=0$ 'daki kazancı da 0dB dir.

$$\text{Geçirme bandı} : A_p \leq 0.707 \quad 0 \leq f \leq 2\text{kHz}=f_p \quad (4.14a)$$

$$\text{Söndürme bandı} : A_s \geq 0.2 \quad 6\text{kHz}=f_s \leq f \leq f_c/2 \quad (4.14b)$$

Alt ve üst kesim frekansları (4.1) eşitliğinde yerine konarak çarpıtma işlemine tabi tutulursa analog domene ait kesim frekansları :

$$f_{ct,p}=2.02.10^3 \text{ Hz} , f_{ct,s}=6.616.10^3 \text{ Hz} \quad (4.15)$$

olarak bulunur. Filtrenin derecesi, (4.2) eşitliğine göre $N=2$ olarak bulunur. Sistemin kesim frekansı, A_p genliğin $1/\sqrt{2}$ 'sine eşit olduğu için $f_{ct,p}$ değerine eşittir. Bu sonuç (4.3) eşitliğinden açıkça görülebilir.

$$T(s) = \frac{1.610877.10^8}{s^2 + 17949.24707s + 1.610877.10^8} \quad (4.16)$$

O halde önceden çarpıtılmış s domeni transfer fonksiyonu, elde edilen bu değerler ve (4.4) eşitlikleri uyarınca (4.16) eşitliğindeki gibi yazılabilir. Daha sonra (4.5) eşitliğinde görülen bilineer dönüşüm uygulanırsa istenen karakteristikleri sağlayan z domeni transfer fonksiyonu :

$$T_{LP}(z) = \frac{0.02427(1+z^{-1})^2}{1-1.5136z^{-1}+0.61z^{-2}} \quad (4.17)$$

olarak elde edilir. Buna göre (3.48) ve (3.49) eşitlikleri gözönüne alınır ve gerekli işlemler yapılırsa Tablo 4.1’de görülen normalize edilmemiş kapasite değerleri elde edilir.

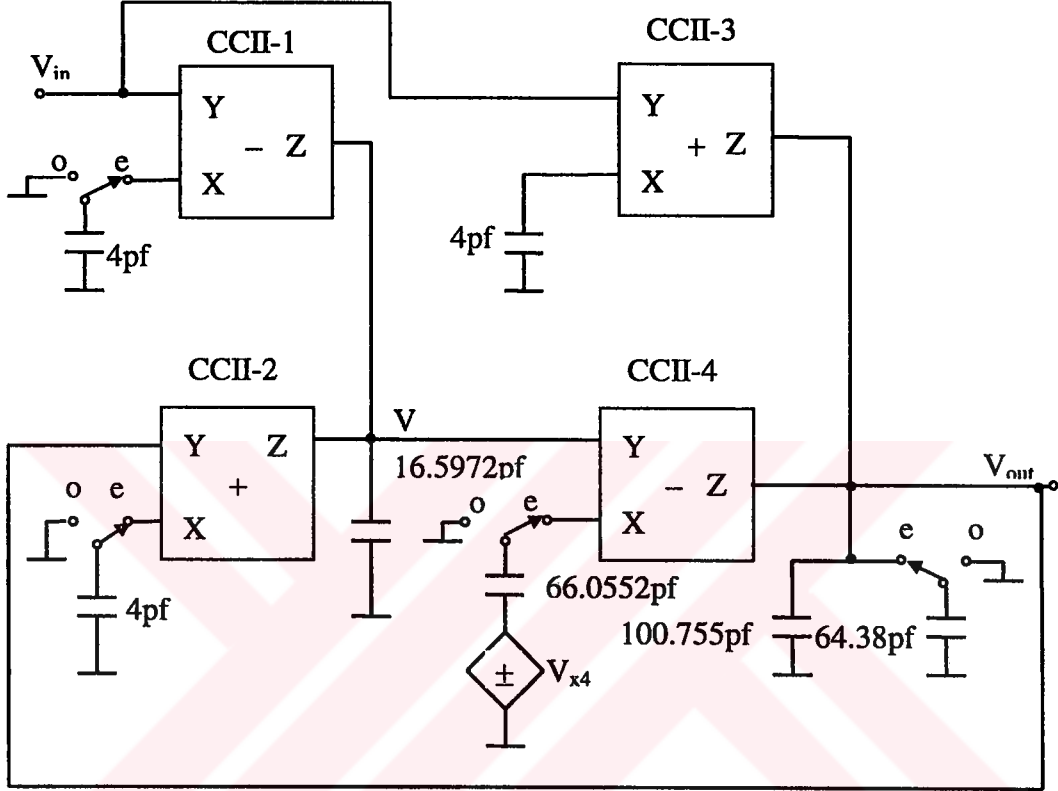
Tablo 4.1 Alçak geçiren filtre gerçekleştirilmesi

Kapasite(pf)	Normalize olmayan değerler	μ Katsayısı ile belirlenen değerler	Son değerler
A	0.09708	0.09708	4
B	0.0964	0.0964	4
C	1	0.4	16.5972
D	1.639	0.6556	66.0552
E	1	1	100.755
F	0.639	0.639	64.38
G	0.0397	0.0397	4

T_{LP} transfer fonksiyonunun maksimum değeri spice simülasyon programı yardımıyla hesaplanırsa $(T_{LP})_{maks} = 1$ olarak bulunur. Aynı işlem T'_{LP} için yapılırsa $(T'_{LP})_{maks} = 0.401287$ olarak bulunur. Bu yüzden (3.46) eşitliklerine göre, $\mu = 2.491982$ olarak hesaplanır. Yine (3.46) eşitlikleri uyarınca C ve D kapasiteleri normalize edilerek Tablo 4.1’de görülen kapasite değerleri elde edilir. Son olarak da A, B, C ve D, E, F, G kapasite grupları, en küçük kapasite 4pf olacak şekilde birbirinden bağımsız olarak normalize edilirler. Bu şekilde devredeki kapasitelerin Tablo 4.1’de görülen son değerleri elde edilir.

Elde edilen son değerlere ilişkin devre şeması Şekil 4.4’de verilmiştir. Bu devrenin girişine, genliği 300mv, periyodu 1kHz (alçak geçiren filtrenin geçirme bandı içinde olmalıdır.), örnekleme frekansı 36kHz olan çift saat fazlarında örneklenmiş bir sinüs işareti uygulanmıştır. Bu devre, spice simülasyon programı yardımıyla analiz edilerek devrenin bu girişe verdiği cevap incelenmiştir[13]. Hem

giriş işaretinin, hem çıkış işaretinin, hem de anahtarları kontrol eden tek ve çift saat fazlı kontrol işaretlerinin spice sonucu elde edilen çıkışları EK-A' da sunulmuştur. Ayrıca devrenin frekans karakteristiği de spice programı yardımıyla çıkartılmış ve EK-A' da sunulmuştur.



Şekil 4.4 İkinci derece alçak geçiren filtre devre topolojisi

4.2.2 Yüksek Geçiren Filtre Devresi

Bir yüksek geçiren filtreye ait örneklenmiş veri domen karakteristikleri (4.18) eşitliklerinde görülmektedir. Bu filtrenin örnekleme zamanı $f_c=36\text{kHz}$ olarak seçilmiştir. Filtrenin dc kazancı 0dB dir.

$$\text{Geçirme bandı} : A_p \leq 0.707 \quad 12\text{kHz} = f_p \leq f \leq 18\text{kHz} = f_c/2 \quad (4.18a)$$

$$\text{Söndürme bandı} : A_s \geq 0.2 \quad 0 \leq f \leq 7\text{kHz} = f_s \quad (4.18b)$$

Tasarlanmak istenen yüksek geçiren filtre öncelikle aynalama yöntemi ile alçak geçiren filtreye dönüştürülür. Bu yapıldığında (4.19)'deki frekans karakteristiği elde edilir.

$$\text{Geçirme bandı} : A_p \leq 0.707 \quad 0 \leq f \leq 6\text{kHz} = f_p \quad (4.19a)$$

$$\text{Söndürme bandı} : A_s \geq 0.2 \quad 11\text{kHz} = f_s \leq f \leq 18\text{kHz} = f_c/2 \quad (4.19b)$$

Elde edilen bu karakteristiklerde görülen alt ve üst kesim frekansları (4.1) eşitliğinde yerine konarak çarpıtma işlemine tabi tutulursa analog domene ait kesim frekansları :

$$f_{ct,p} = 6.616 \cdot 10^3 \text{ Hz}, \quad f_{ct,s} = 16.36537 \cdot 10^3 \text{ Hz} \quad (4.20)$$

olarak bulunur. Filtrenin derecesi (4.2) eşitliğine göre $N=2$ olarak bulunur. Sistemin kesim frekansı, A_p genliğin $1/\sqrt{2}$ 'sine eşit olduğu için $f_{ct,p}$ değerine eşittir. Bu sonuç (4.3) eşitliğinden açıkça görülebilir. O halde önceden çarpıtılmış s domeni transfer fonksiyonu, bu sonuçlar ve (4.4) eşitlikleri uyarınca :

$$T(s) = \frac{1.7280278 \cdot 10^9}{s^2 + 58788.277s + 1.7280278 \cdot 10^9} \quad (4.21)$$

olarak elde edilir. Daha sonra bilineer dönüşüm uygulanırsa alçak geçiren filtrenin karakteristiklerini sağlayan z domeni transfer fonksiyonu:

$$T_{LP}(z) = \frac{0.155(1+z^{-1})^2}{1-0.62z^{-1}+0.24z^{-2}} \quad (4.22)$$

olarak elde edilir. Bu transfer fonksiyonunda z yerine -z yazılırsa tasarlanmak istenen yüksek geçiren filtre transfer fonksiyonu (4.23)'de görüldüğü gibi elde edilir.

$$T_{LP}(z) = \frac{0.155(1 - z^{-1})^2}{1 + 0.62z^{-1} + 0.24z^{-2}} \quad (4.23)$$

Buna göre (3.52) ve (3.53) eşitlikleri gözönüne alınır ve gerekli işlemler yapılırsa Tablo 4.2’de görülen normalize edilmemiş kapasite değerleri elde edilir.

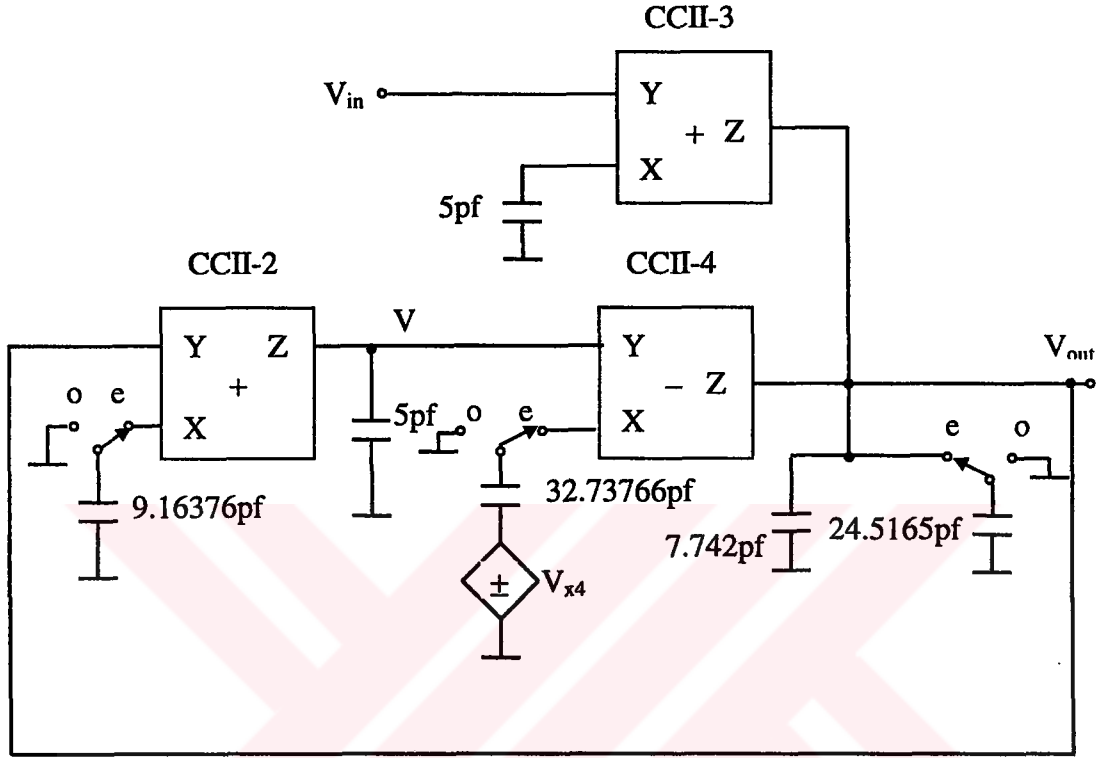
Tablo 4.2 Yüksek geçiren filtre gerçekleştirilmesi

Kapasite(pf)	Normalize olmayan değerler	μ Katsayısı ile belirlenen değerler	Son değerler
B	1.86	1.86	9.16376
C	1	1.0152	5
D	4.1667	4.23	32.73766
E	1	1	7.742
F	3.1667	3.1667	24.5165
G	0.64583	0.64583	5

T_{HP} transfer fonksiyonunun maksimum değeri spice simülasyon programı yardımıyla hesaplanırsa $(T_{HP})_{maks}=1$ olarak bulunur. Aynı işlem T'_{HP} için yapılırsa $(T'_{HP})_{maks}=1.0152$ olarak bulunur. Bu yüzden (3.46) eşitliklerine göre, $\mu=0.985026$ olarak hesaplanır. Yine (3.46) eşitlikleri uyarınca C ve D kapasiteleri normalize edilerek Tablo 4.2’de görülen kapasite değerleri elde edilir. Son olarak da B, C ve D, E, F, G kapasite grupları, en küçük kapasite 5pf olacak şekilde birbirinden bağımsız olarak normalize edilirler. Bu şekilde devredeki kapasitelerin Tablo 4.2’de görülen son değerleri elde edilmiş olur.

Elde edilen son değerlere ilişkin devre şeması Şekil 4.5’de verilmiştir. Bu devrenin girişine, genliği 200mv, periyodu 15kHz (yüksek geçiren filtrenin geçirme bandı içinde olmalıdır.), örnekleme frekansı 36kHz olan çift saat fazlarında örneklenmiş bir sinüs işareti uygulanmıştır. Bu devre, spice simülasyon programı yardımıyla analiz edilerek devrenin bu girişe verdiği cevap incelenmiştir[13]. Hem giriş işaretinin, hem çıkış işaretinin, hem de anahtarları kontrol eden tek ve çift saat

fazlı kontrol işaretlerinin spice sonucu elde edilen çıkışları EK-B' de sunulmuştur. Ayrıca devrenin frekans karakteristiği de spice programı yardımıyla çıkartılmış ve EK-B'de sunulmuştur.



Şekil 4.5 İkinci derece yüksek geçiren filtre topolojisi

4.2.3 Band Geçiren Filtre Devresi

Tasarlanmak istenen ikinci dereceden band geçiren filtrenin merkez frekansı $f_o=1633\text{Hz}$, merkez frekansındaki kazancı 10dB, iyilik katsayısı $Q=16$ dır. Örnekleme frekansı $f_c=8\text{kHz}$ olarak seçilmiştir. İyilik katsayısı $Q \geq 10$ şartını sağladığından alt ve üst kesim frekansları (4.24) eşitliklerinde görüldüğü gibi hesaplanır.

$$f_L = f_o \left(1 - \frac{1}{2Q} \right) = 1581.968\text{Hz} \quad (4.24a)$$

$$f_H = f_o \left(1 + \frac{1}{2Q} \right) = 1684.03125 \text{ Hz} \quad (4.24b)$$

Merkez frekansı, alt ve üst kesim frekansları (4.1) eşitliğinde yerine konarak çarpıtma işlemine tabi tutulursa yeni merkez frekansı $f_{ct, o} = 1901 \text{ Hz}$, yeni kesim frekansları da $f_{ct, L} = 1822.71562 \text{ Hz}$, $f_{ct, H} = 1981.716366 \text{ Hz}$ olarak bulunur. Elde edilen bu veriler (4.12) ve (4.13) eşitliklerinde yerine konur ve gerekli düzenlemeler yapılırsa (4.25)'de görülen önceden çarpıtılmış s domeni transfer fonksiyonu elde edilir.

$$T(s) = \frac{999.03s}{s^2 + 999.03s + 1.428 \cdot 10^8} \quad (4.25)$$

Bu transfer fonksiyonuna bilineer dönüşüm uygulanırsa istenen karakteristikleri sağlayan z domeni ikinci dereceden band geçiren filtre transfer fonksiyonu (4.26)'da görüldüğü gibi elde edilir.

$$T_{BP}(z) = \frac{0.1219(1+z^{-1})(1-z^{-1})}{1-0.5458z^{-1}+0.9229z^{-2}} \quad (4.26)$$

Buna göre (3.56) ve (3.57) eşitlikleri gözönüne alınır ve gerekli işlemler yapılırsa Tablo 4.3'de görülen normalize edilmemiş kapasite değerleri elde edilir. T_{BP} transfer fonksiyonunun maksimum değeri spice simülasyon programı yardımıyla hesaplanırsa $(T_{BP})_{maks} = 3.162233$ olarak bulunur. Aynı işlem T'_{BP} için yapılırsa $(T'_{BP})_{maks} = 3.622438$ olarak bulunur. Bu yüzden (3.46) eşitliklerine göre, $\mu = 0.872957$ olarak hesaplanır. Yine (3.46) eşitlikleri uyarınca C ve D kapasiteleri normalize edilerek Tablo 4.3'de görülen kapasite değerleri elde edilir. Son olarak da B, C ve D, E, F, G, H kapasite grupları, en küçük kapasite 4pf olacak şekilde birbirinden bağımsız olarak normalize edilirler. Bu şekilde devredeki kapasitelerin Tablo 4.3'de görülen son değerleri elde edilmiş olur. Elde edilen son değerlere ilişkin devre şeması Şekil 4.6'da verilmiştir. Bu devrenin girişine, genliği 300mv, periyodu 1633Hz (Band geçiren filtrenin geçirme bandı merkez frekansı), örnekleme frekansı

8kHz olan çift saat fazlarında örneklenmiş bir sinüs işareti uygulanmıştır.

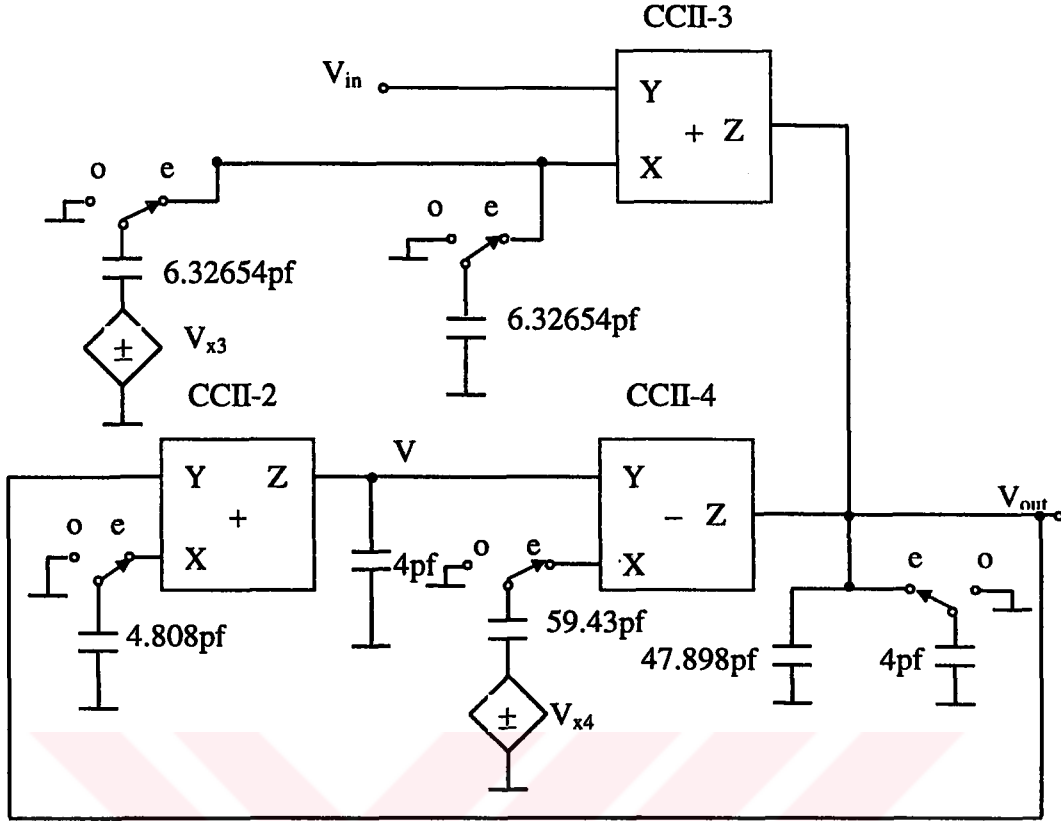
Tablo 4.3 Band geçiren filtre gerçekleştirilmesi

Kapasite(pf)	Normalize olmayan Değerler	μ Katsayısı ile belirlenen değerler	Son değerler
B	1.3771	1.3771	4,808
C	1	1.145531	4
D	1.083541	1.24123	59.43
E	1	1	47.898
F	0.083541	0.083541	4
G	0.132	0.132	6.32654
H	0.132	0.132	6.32654

Bu devre, spice simülasyon programı yardımıyla analiz edilerek devrenin bu girişe verdiği cevap incelenmiştir[13]. Hem giriş işaretinin, hem çıkış işaretinin, hem de anahtarları kontrol eden tek ve çift saat fazlı kontrol işaretlerinin spice sonucu elde edilen çıkışları EK-C' de sunulmuştur. Ayrıca devrenin frekans karakteristiği de spice programı yardımıyla çıkartılmış ve EK-C' de sunulmuştur Yukarıda tasarlanan band geçiren filtre transfer fonksiyonunun $z = -1$ 'de bir sıfırı vardır. Bu sıfır kaldırılarak elde edilen transfer fonksiyonu da bir band geçiren filtre gerçekler. Bu sıfır kaldırıldığında elde edilecek transfer fonksiyonu (4.27)'de görülmektedir.

$$T_{BP}(z) = \frac{0.1953(1 - z^{-1})}{1 - 0.5458z^{-1} + 0.9229z^{-2}} \quad (4.27)$$

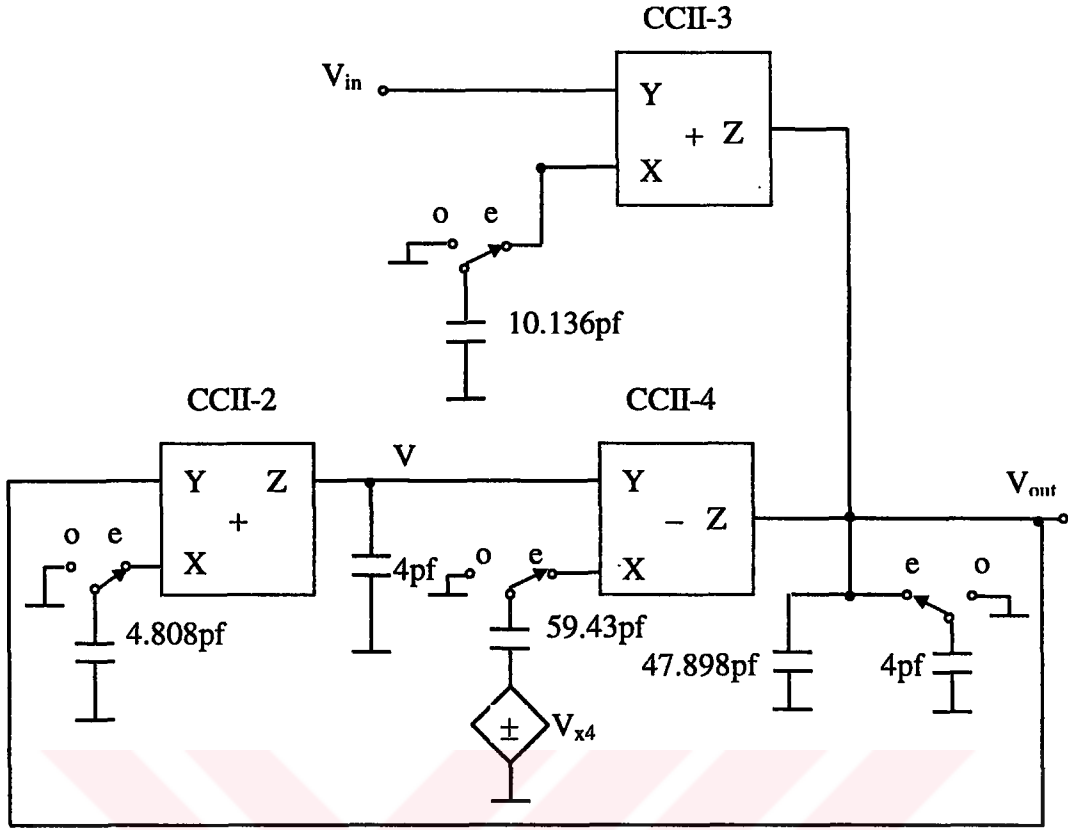
Her iki transfer fonksiyonunun kazancının 10 dB olması için ikinci transfer fonksiyonun kazanç sabiti(K) artırılmıştır. Sıfırın atılması, H kapasitesinin devre dışı kalması sonucunu doğurmuştur. Aynı zamanda G kapasitesinin değeri, kazanç sabiti artışı için 10.136pf değerine çıkmıştır. Bu iki fark haricinde devre topolojileri aynıdır. İkinci transfer fonksiyonuna ait devre Şekil 4.7'de verilmiştir.



Şekil 4.6 İkinci derece band geçiren aktif filtre topolojisi

Her iki devrenin z domeni frekans eğrileri, spice simülasyon programı yardımıyla incelemiş ve EK-C'de sunulmuştur. İki eğri arasındaki en temel fark, $z = -1$ 'de sıfırı olan band geçiren filtre transfer fonksiyonunun, 4kHz civarında keskin bir sıfır geçişine sahip olmasıdır. $z = -1$ 'deki bu hata sadece örnekleme frekansının arttırılmasıyla ihmal edilebilir bir seviyeye çekilebilir. Ayrıca her iki devrenin, geçirme bandı içinde uygulanan örneklenmiş giriş işaretine karşı verdiği cevaplar spice simülasyon programı yardımıyla incelendiğinde birbirlerinden çok farklı olmadığı görülmüştür. Elde edilen bu çıkış işaretlerine ilişkin spice sonuçları da EK-C'de sunulmuştur.

Sonuç olarak, $z = -1$ 'deki sıfırın atılmasıyla elde edilen devreye ait toplam kapasite miktarının diğerine göre daha az olması ve frekans eğrilerinin geçirme bandı içinde kalan kısımlarının hemen hemen aynı olması, $z = -1$ 'deki sıfırın atılmasıyla elde edilen devreyi daha avantajlı kılar.



Şekil 4.7 $z = -1$ 'deki sıfırı atılarak elde edilen ikinci derece band geçiren aktif filtre topolojisi

4.2.4 Üçüncü Dereceden Alçak Geçiren Filtre Devresi

$$\text{Geçirme bandı} : A_p \leq 0.707 \quad 0 \leq f \leq 2\text{kHz} = f_p \quad (4.28a)$$

$$\text{Söndürme bandı} : A_s \geq 0.2 \quad 4\text{kHz} = f_s \leq f \leq f_c/2 \quad (4.28b)$$

Örneklenmiş veri domeni karakteristikleri (4.28) eşitliklerinde verilen, alçak geçiren filtrenin örnekleme zamanı $f_c = 36\text{kHz}$, $\omega = 0$ 'daki kazancı 0dB dir. Gerekli ara işlemler yapıldığında bu filtreye ait z domeni transfer fonksiyonu (4.29) eşitliğindeki gibi elde edilir.

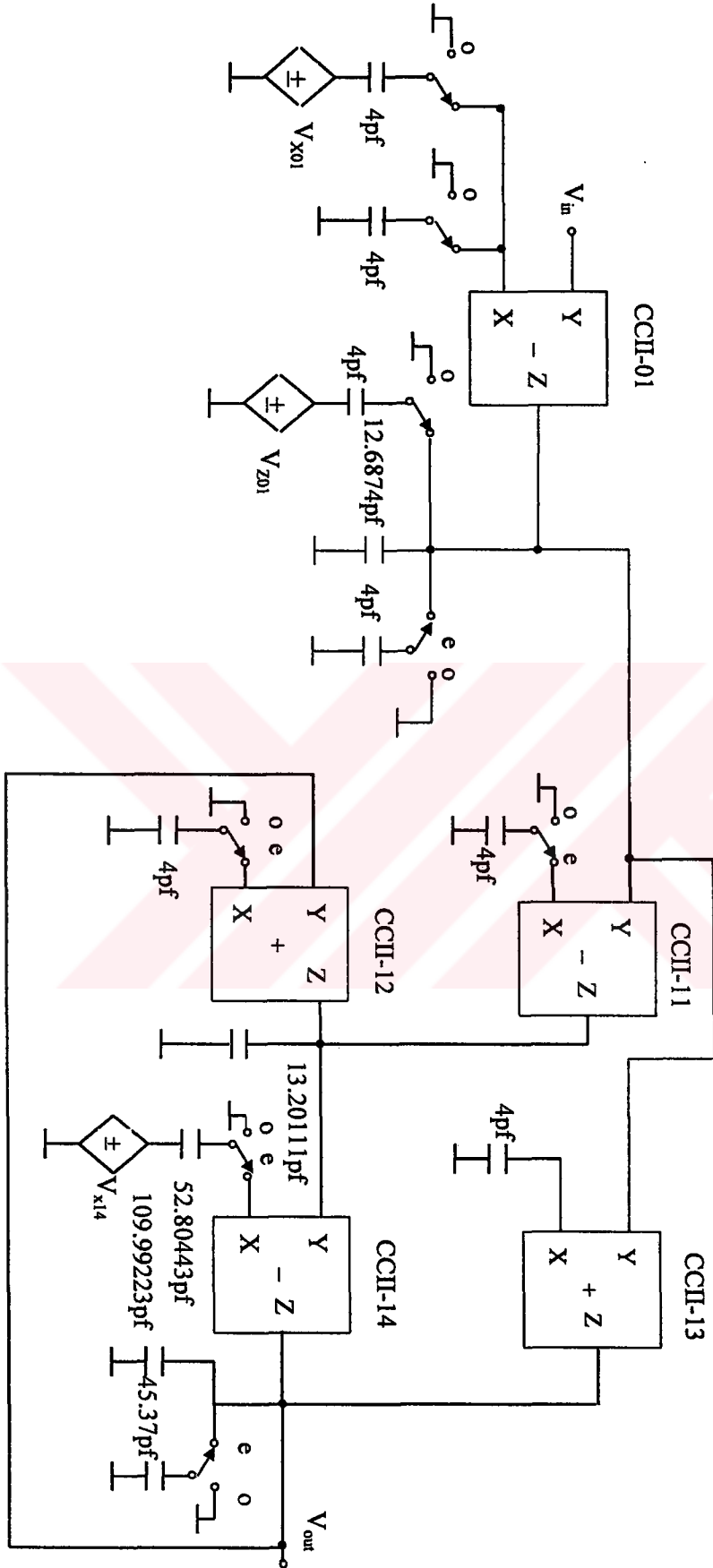
$$T_{LP}(z) = \frac{0.149883(1+z^{-1})}{1-0.700233z^{-1}} \cdot \frac{0.025745(1+z^{-1})^2}{1-1.604969z^{-1}+0.707951z^{-2}} \quad (4.29)$$

(4.29) eşitliğinde görülen transfer fonksiyonu birinci ve ikinci dereceden iki çarpandan oluşmaktadır. Birinci çarpan bir integratör devresi, ikinci çarpan ise bölüm 3’de ayrıntılarıyla verilen ikinci derece alçak geçiren blok yapı ile gerçekleşir. Bu iki yapının kaskad bağlanması ile (4.29)’da görülen üçüncü dereceden alçak geçiren filtre devresi gerçekleştirilmiş olur. Bu yapıldığında Şekil 4.8’deki devre yapısı elde edilir.

Tablo 4.4 Üçüncü Dereceden Alçak Geçiren Filtre Gerçeklemesi

Birinci Kata İlişkin Elemanlar			
Kapasite(pf)	Normalize olmayan Değerler		Son değerler
C1	1		4
C2	5.671848469		22.6874
C3	1		4
İkinci Kata İlişkin Elemanlar			
Kapasite(pf)	Normalize olmayan Değerler	μ Katsayısı ile belirlenen değerler	Son değerler
A	0.10298	0.10298	4
B	0.10298	0.10298	4
C	1	0.33987	13.20111
D	1.41253	0.48007	52.80443
E	1	1	109.99223
F	0.41253	0.41253	45.37482
G	0.03637	0.03637	4

İntegratör yapısına ait devre elemanları (3.28), ikinci derece yapıya ait devre elemanları (3.52) ve (3.53) eşitlikleri yardımıyla bulunabilir. Buna göre gerekli işlemler yapıldığında üçüncü dereceden alçak geçiren filtre devresine ait devre elemanlarının normalize olmayan değerleri Tablo 4.4’de görüldüğü gibi elde edilir. İntegratör devresi için en küçük kapasite değeri 4pf olacak şekilde normalize edilirse Tablo 4.4’de görülen son değerler bulunur. İkinci derece yapıya ilişkin eleman



Şekil 4.8 Üçüncü Dereceden Alçak Geçiren Filtre Devresi

değerlerini normalize edebilmek için de $(T_{LP})_{maks}$ ve $(T'_{LP})_{maks}$ değerlerinin bulunması gereklidir. Bu işlem spice simülasyon programı yardımıyla yapılırsa $(T_{LP})_{maks}=1$, $(T'_{LP})_{maks}=0.339869$ olarak bulunur. Buradan $\mu=2.942311$ olarak tespit edilir. Bu değerlerin ışığında ikinci derece yapıya ilişkin elemanların son değerleri yani normalize edilmiş değerleri yine Tablo 4.4'de görüldüğü gibi elde edilir.

Bu devrenin girişine, genliği 300mv, periyodu 1kHz (alçak geçiren filtrenin geçirme bandı içinde olmalıdır.), örnekleme frekansı 36kHz olan çift saat fazlarında örneklenmiş bir sinüs işareti uygulanmıştır. Bu devre, spice simülasyon programı yardımıyla analiz edilerek devrenin bu giriş verdiği cevap incelenmiştir[13]. Hem giriş işaretinin, hem çıkış işaretinin, hem de anahtarları kontrol eden tek ve çift saat fazlı kontrol işaretlerinin spice sonucu elde edilen çıkışları EK-D' de sunulmuştur. Ayrıca devrenin frekans karakteristiği de spice programı yardımıyla çıkartılmış ve EK-D' de sunulmuştur.

4.2.5 Dördüncü Dereceden Band Geçiren Filtre Devresi

Bu bölümde alt kesim frekansı 7142.646141Hz, üst kesim frekansı 9623Hz, merkez frekansındaki kazancı 0dB, örnekleme frekansı 40kHz olan dördüncü dereceden band geçiren filtre tasarlanacaktır. Bunun için bölüm 4.1'de anlatılan yöntemler kullanılır ve gerekli işlemler yapılırsa dördüncü dereceden band geçiren filtrenin z domen transfer fonksiyonu (4.30) eşitliğinde görüldüğü gibi elde edilir.

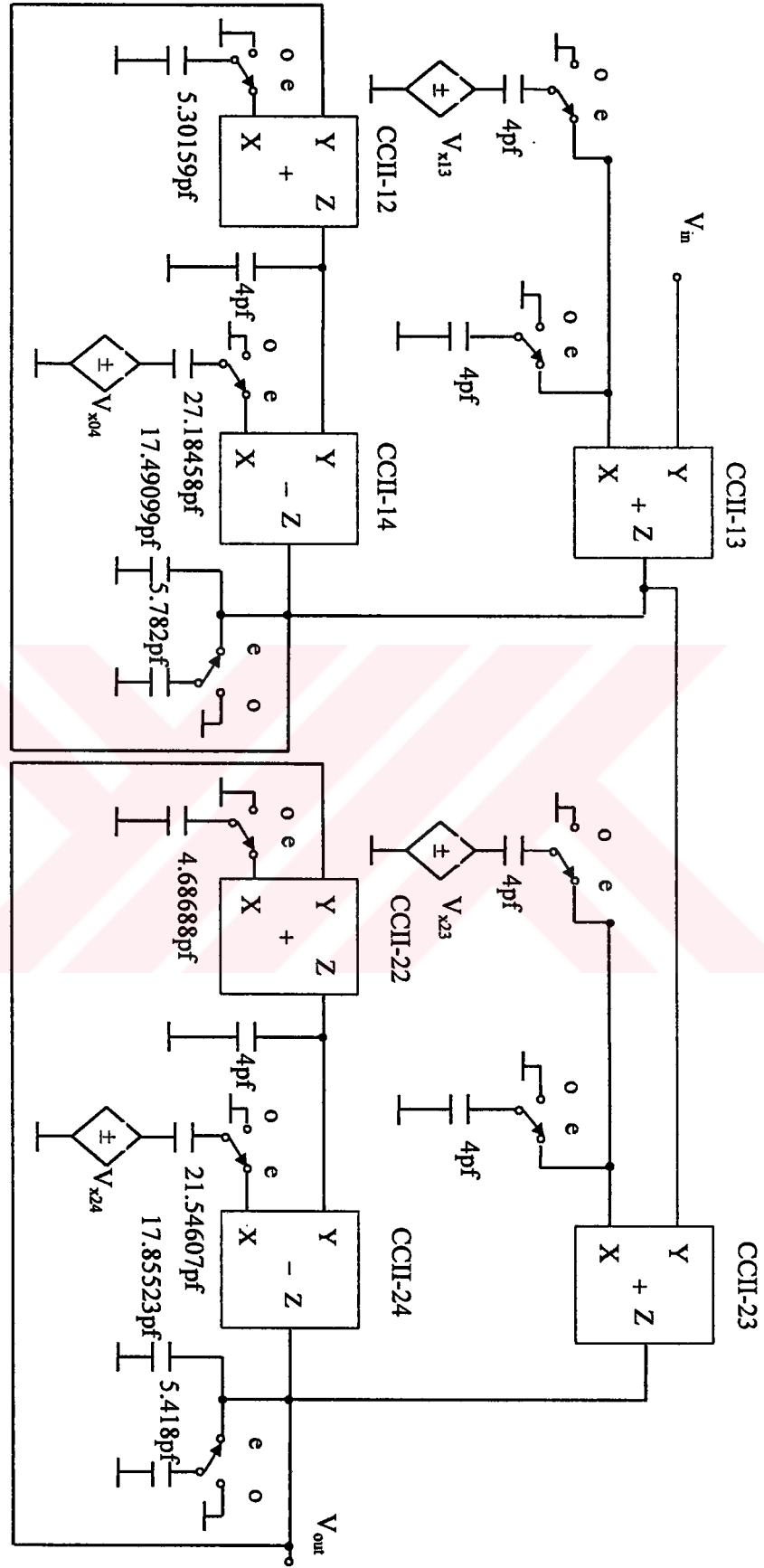
$$T_{BP}(z) = \frac{0.172013761(1+z^{-1})(1-z^{-1})}{1-0.2038z^{-1}+0.751384z^{-2}} \frac{0.172013761(1+z^{-1})(1-z^{-1})}{1-0.68325z^{-1}+0.767034z^{-2}} \quad (4.30)$$

Görüldüğü gibi transfer fonksiyonu iki tane ikinci dereceden band geçiren transfer fonksiyonunun çarpımı şeklindedir. Bu nedenle bu fonksiyon, iki tane ikinci dereceden band geçiren filtre devresinin birbirine kaskad bağlanmasıyla gerçekleştirilebilir. Bu yapıldığında Şekil 4.9'daki devre elde edilir. Birinci ve ikinci kata ilişkin devre elemanlarının değerleri (3.56) ve (3.57) eşitlikleri yardımıyla bulunabilir. Bu yapıldığında Tablo 4.5'de görülen normalize edilmemiş kapasite değerlerine ulaşılır.

Her iki kata ilişkin normalize edilmiş eleman değerlerinin bulunabilmesi için öncelikle her katın T_{maks} , T'_{maks} ve μ değerlerinin tespit edilmesi gereklidir. Bu işlem spice simülasyon programı yardımıyla yapılacak olursa birinci kata ilişkin $T_{maks}=1.383370$, $T'_{maks}=1.615827$, $\mu_1=0.856137$, ikinci kata ilişkin $T_{maks}=1$, $T'_{maks}=0.925767$, $\mu_2=1.080184$, olarak bulunur. Bu değerler ve (3.46) eşitlikleri gözönüne alınarak, en küçük kapasite değeri 4pf olacak şekilde gerekli düzenlemeler yapılırsa devreye ilişkin normalize edilmiş kapasite değerleri Tablo 4.5'de "son değerler" sütununda görüldüğü gibi elde edilir.

Tablo 4.5 Dördüncü dereceden band geçiren filtre gerçekleştirilmesi

Birinci Kata İlişkin Eleman Değerleri			
Kapasite(pf)	Normalize olmayan Değerler	μ Katsayısı ile belirlenen değerler	Son değerler
B	1.54811	1.54811	5.30159
C	1	1.16804	4
D	1.33061	1.55420	27.18458
E	1	1	17.49099
F	0.33061	0.33061	5.78273
G	0.22869	0.22869	4
H	0.22869	0.22869	4
İkinci Kata İlişkin Eleman Değerleri			
Kapasite(pf)	Normalize olmayan Değerler	μ Katsayısı ile belirlenen değerler	Son değerler
B	1.08477	1.08474	4.68688
C	1	0.92577	4
D	1.30347	1.20671	21.54607
E	1	1	17.85523
F	0.30347	0.30347	5.41849
G	0.22402	0.22402	4
H	0.22402	0.22402	4



Şekil 4.9 Dördüncü Dereceden Band Geçiren Filtre Devresi

Bu devrenin girişine, genliği 250mv, periyodu 8.346kHz (band geçiren filtrenin geçirme bandı içinde olmalıdır.), örnekleme frekansı 40kHz olan çift saat fazlarında örneklenmiş bir sinüs işareti uygulanmıştır. Bu devre, spice simülasyon programı yardımıyla analiz edilerek devrenin bu girişe verdiği cevap incelenmiştir[13]. Hem giriş işaretinin, hem çıkış işaretinin, hem de anahtarları kontrol eden tek ve çift saat fazlı kontrol işaretlerinin spice sonucu elde edilen çıkışları EK-E' de sunulmuştur. Ayrıca devrenin frekans karakteristiği de spice programı yardımıyla çıkartılmış ve EK-E' de sunulmuştur.



BÖLÜM 5

AKIM TAŞIYICI İÇEREN SC AKTİF FİLTRELERİN BİLGİSAYAR PROGRAMI İLE TASARIMI

Akım taşıyıcı SC aktif filtrelerin ve z domeni filtre fonksiyonlarının tasarım ilkeleri bilgisayar ortamına aktararak, akım taşıyıcı SC aktif filtrelerin bilgisayar programı yardımı ile tasarlanması amacıyla C dilinde bir bilgisayar programı hazırlanmıştır. Hazırlanan program alçak, yüksek ve band geçiren filtre fonksiyonlarının tasarımını yapabilmektedir. Bu program temel olarak üç ana bölümden oluşur. Birinci bölüm, istenen özellikleri sağlayan alçak, yüksek, band geçiren filtrelere ilişkin z domeni transfer fonksiyonlarının tasarımını içerir. İkinci bölüm, bu fonksiyonların birinci ve ikinci dereceden fonksiyonlara parçalanarak, herbirinin katsayıları uyarınca herbirine ilişkin normalize edilmiş kapasite değerlerinin hesaplanmasını içerir. Üçüncü bölüm ise, birinci bölümde hesaplanan filtre transfer fonksiyonlarına ait frekans karakteristiklerinin çizdirilmesini içerir.

Bu program, filtre transfer fonksiyonlarının hesaplanmasında bilineer dönüşüm yöntemini kullanmaktadır. Alçak geçiren filtre tasarımı için girilen örneklenmiş veri domenine ait frekans parametreleri, (5.1) eşitliği kullanılarak analog domene çevrilir.

$$\omega_{\alpha} = \frac{2}{\tau} \tan\left(\frac{\omega\tau}{2}\right) \quad (5.1)$$

(5.1) eşitliğinde görülen τ , örnekleme zamanını simgelemektedir. Analog domene çevrilen bu frekans değerleri, (5.2) ve (5.3) eşitliklerinde yerine konarak filtrenin derecesi ve kesim frekansı hesaplanır. Derecesi ve kesim frekansı bu şekilde hesaplandıktan sonra (5.4) eşitlikleri uyarınca s domeni filtre transfer fonksiyonları program tarafından bulunur.

$$\omega_{\text{cutoff}} = \frac{\omega_{\text{ct,p}}}{\left(\frac{1}{A_p^2} - 1\right)^{1/2N}} \quad (5.2)$$

$$N \geq \frac{1}{2} \frac{\log \left[\left(\frac{1}{A_s^2} - 1 \right) / \left(\frac{1}{A_p^2} - 1 \right) \right]}{\log \left[\frac{\omega_{\text{ct,s}}}{\omega_{\text{ct,p}}} \right]} \quad (5.3)$$

$$T(s) = \prod_{k=1}^{N/2} \frac{\omega_{\text{cutoff}}^2}{s^2 + \omega_{\text{cutoff}}^2 b_k s + \omega_{\text{cutoff}}^2}; \quad N \text{ çift} \quad (5.4a)$$

$$T(s) = \frac{\omega_{\text{cutoff}}}{s + \omega_{\text{cutoff}}} \prod_{k=1}^{(N/2)-1} \frac{\omega_{\text{cutoff}}^2}{s^2 + \omega_{\text{cutoff}}^2 b_k s + \omega_{\text{cutoff}}^2}; \quad N \text{ tek} \quad (5.4b)$$

$$b_k = 2 \sin \left[(2k-1) \frac{\pi}{2N} \right] \quad (5.4c)$$

Bu şekilde elde edilen transfer fonksiyonuna bilineer dönüşüm uygulanarak z domeni transfer fonksiyonu hesaplanır. Bunun için s yerine (5.5)' de görülen ifade yerleştirilir.

$$s \rightarrow \frac{2}{\tau} \frac{1-z^{-1}}{1+z^{-1}} = 2f_c \frac{1-z^{-1}}{1+z^{-1}} \quad (5.5)$$

Yüksek geçiren transfer fonksiyonunu hesaplamak için dijital özellikleri, aynalama yöntemini kullanarak alçak geçiren filtre karakteristiğine çevirir. Alçak geçiren filtre tasarımında anlatılan adımlar tek tek uygulanarak alçak geçiren transfer fonksiyonu program tarafından hesaplanır. Bu fonksiyonda z yerine $-z$ konarak yüksek geçiren filtre transfer fonksiyonu elde edilir ve ekrana yazılır.

Band geçiren filtre transfer fonksiyonun tasarımında ise bu fonksiyonun derecesinin yarı derecesine sahip s domeni butterworth alçak geçiren transfer

fonksiyonuna (5.6)' da görülen dönüşüm kuralı uygulanır. Daha sonra elde edilen fonksiyona (5.5) eşitliğinde görülen bilineer dönüşüm uygulanarak z domeni band geçiren filtre fonksiyonu hesaplanır.

$$s \rightarrow \frac{s^2 + \omega_{ct,o}^2}{(\omega_{ct,H} - \omega_{ct,L})s} \quad (5.6)$$

Bulunan bu fonksiyonlar program tarafından birinci ve ikinci dereceden fonksiyonlara parçalanır. Herbir fonksiyonun katsayıları cinsinden devrede yer alan kapasite değerleri, üçüncü bölümde ayrıntılarıyla açıklanan tasarım eşitliklerine göre hesaplanır. Daha sonra yine üçüncü bölümde açıklanan normalizasyon metoduna göre T ve T' fonksiyonlarının maksimum değerleri herbir kat için hesaplanarak μ normalizasyon katsayısı program tarafından bulunur. Bu katsayı ve kullanıcı tarafından parametre olarak girilen minimum kapasite değeri dikkate alınarak normalize edilmiş kapasite değerleri herbir kat için ayrı ayrı hesaplanır ve ekrana yazılır.

Alçak ve yüksek geçiren filtre transfer fonksiyonlarından 200, band geçiren filtre transfer fonksiyonlarından da 400 nokta alınarak bu fonksiyonlara ilişkin frekans karakteristikleri, logaritmik eksene göre çizilerek ekranda görüntülenir.

ANAMENU				
İ	.[F1].....	AGS .	İ	
	.[F2].....	YGS .		
T	.[F3].....	BGS .	T	
	.[F4].....	YARDIM.		
Ü	.[F5].....	ÇIKIŞ .	Ü	

Şekil 5.1 Program çalıştırıldığında karşılaşılabilecek ilk menü ekranı

Buraya kadar programın hazırlanma amacı ve programın matematiksel olarak izlediği yöntem üzerinde durulmuştur. Şimdi de program çalıştırıldığında karşılaşılabilecek menü ekranları ve verdiği sonuçlar incelenecektir.

Program çalıştırıldığında karşılaşılabilecek ilk menü ekranı, Şekil 5.1’de görülmektedir. Bu şekilden de görüldüğü gibi beş seçenek vardır. Bu seçeneklerden herhangi birini seçmek için karşısında görülen tuşa basmak yeterlidir. Örneğin, “F1” tuşuna basıldığında parametre menüsü olarak isimlendirilen ikinci bir menü ile karşılaşılır. Bu menü ekranı Şekil 5.2’de görülmektedir.

LÜTFEN DİJİTAL ÖZELLİKLERİ GİRİNİZ		F7 YÜKLEME
A_p	=	F8 SAKLAMA
A_s	=	F9 HESAPLAMA
F_p (Hz)	=	F10 ANA MENÜ
F_s (Hz)	=	DEL SATIR SİLME
F_c (Hz)	=	
K_v (dB)	=	
MKD (pF)	=	

Şekil 5.2 Parametre menüsü

Şekil 5.2’de görülen menüde F_p , kesim frekansını, F_s , durdurma bandı frekansını, A_p , F_p frekansındaki genliği, A_s , F_s frekansındaki genliği, F_c , örnekleme frekansını, K_v , dB olarak filtre kazancını, MKD, devrede yer alması istenen minimum kapasite değerini simgeler. Örneğin $A_p=0.707$, $A_s=0.2$, $F_p=2000$, $F_s=6000$, $F_c=36000$, $K_v=0$, MKD=4 olarak girilirse Şekil 5.3’de görülen transfer fonksiyonu ve devreye ilişkin kapasite listesi elde edilir. Bu transfer fonksiyonuna ilişkin frekans eğrisini, görmek için “F2” tuşuna basıldığında Şekil 5.4’de görülen eğri elde edilir.

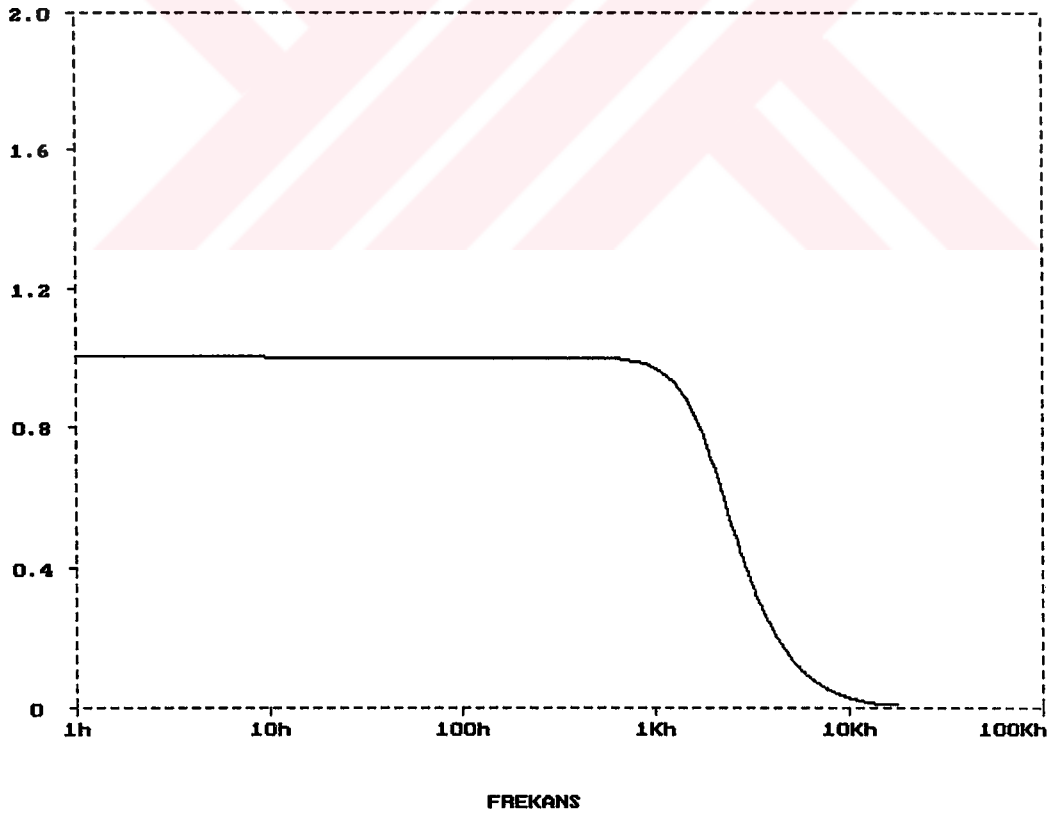
Anamenüden yüksek geçiren filtre seçeneği seçilirse, Şekil 5.2’de görülen parametre menüsü ekrana gelir. Burada $A_p=0.707$, $A_s=0.2$, $F_p=12000$, $F_s=7000$, $F_c=36000$, $K_v=0$, MKD=5 olarak girilirse Şekil 5.5’de görülen transfer fonksiyonu ve devrede yer alan kapasite değerleri elde edilir. Bu transfer fonksiyonunun frekans eğrisi ise Şekil 5.6’da görülmektedir.

$$H(Z) = \prod_{j=1}^N H_j(Z) \quad ; \quad N=2. \text{ DERECE AGS}$$

$$H_1(Z) = \frac{0.024275(1+Z^{-1})^2}{1 - 1.513452Z^{-1} + 0.610552Z^{-2}}$$

$C[1][1] = 4.00000\text{pf}$
 $C[1][2] = 4.00000\text{pf}$
 $C[1][3] = 16.57144\text{pf}$
 $C[1][4] = 4.00000\text{pf}$
 $C[1][5] = 66.28577\text{pf}$
 $C[1][6] = 100.60613\text{pf}$
 $C[1][7] = 64.17291\text{pf}$

Şekil 5.3 Program tarafından hesaplanan alçak geçiren filtre fonksiyonu
Ve bu fonksiyona ilişkin kapasite değerleri



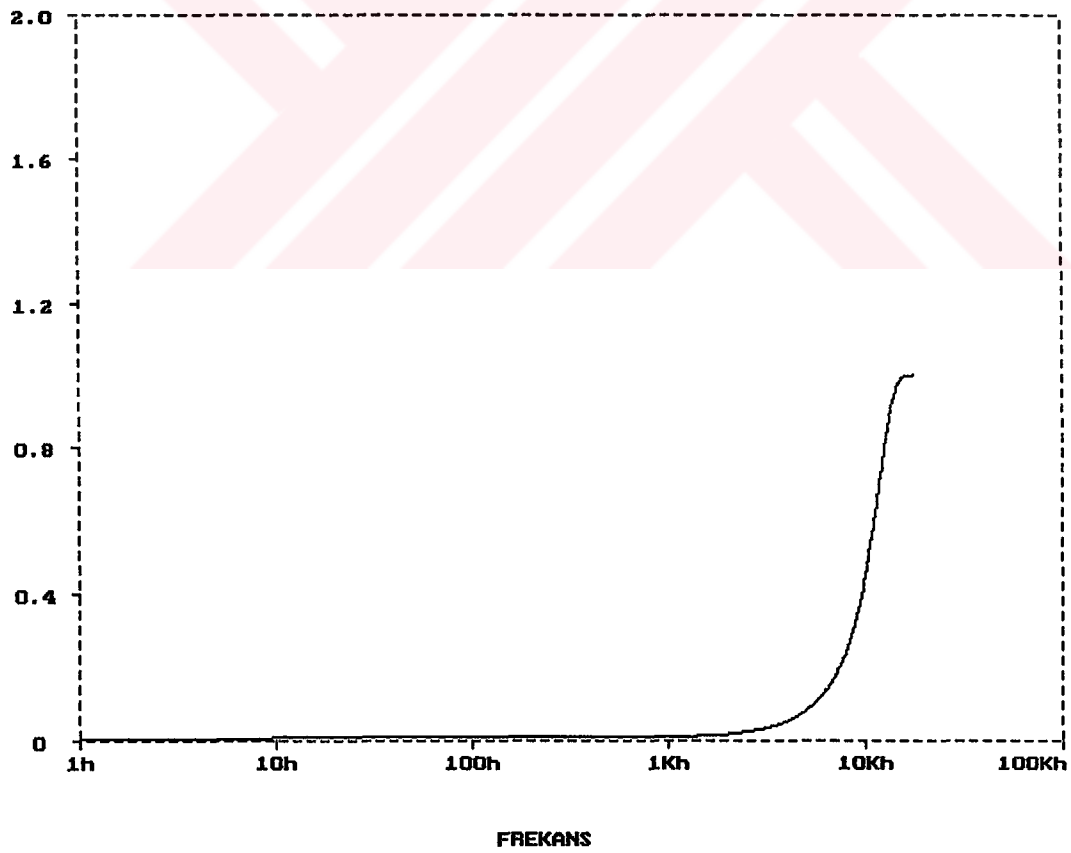
Şekil 5.4 Şekil 5.3’de görülen transfer fonksiyonuna ilişkin frekans eğrisi

$$H(Z) = \prod_{j=1}^N H_j(Z) ; N=2. \text{ DERECE HGS}$$

$$H_1(Z) = \frac{0.155020(1-Z^{-1})^2}{1 + 0.620362Z^{-1} + 0.240444Z^{-2}}$$

$C[1][1] = 5.00000\text{pf}$
 $C[1][2] = 9.86753\text{pf}$
 $C[1][3] = 5.00000\text{pf}$
 $C[1][4] = 30.41191\text{pf}$
 $C[1][5] = 7.75523\text{pf}$
 $C[1][6] = 24.49860\text{pf}$

Şekil 5.5 Program tarafından hesaplanan yüksek geçiren filtre fonksiyonu
Ve bu fonksiyona ilişkin kapasite değerleri



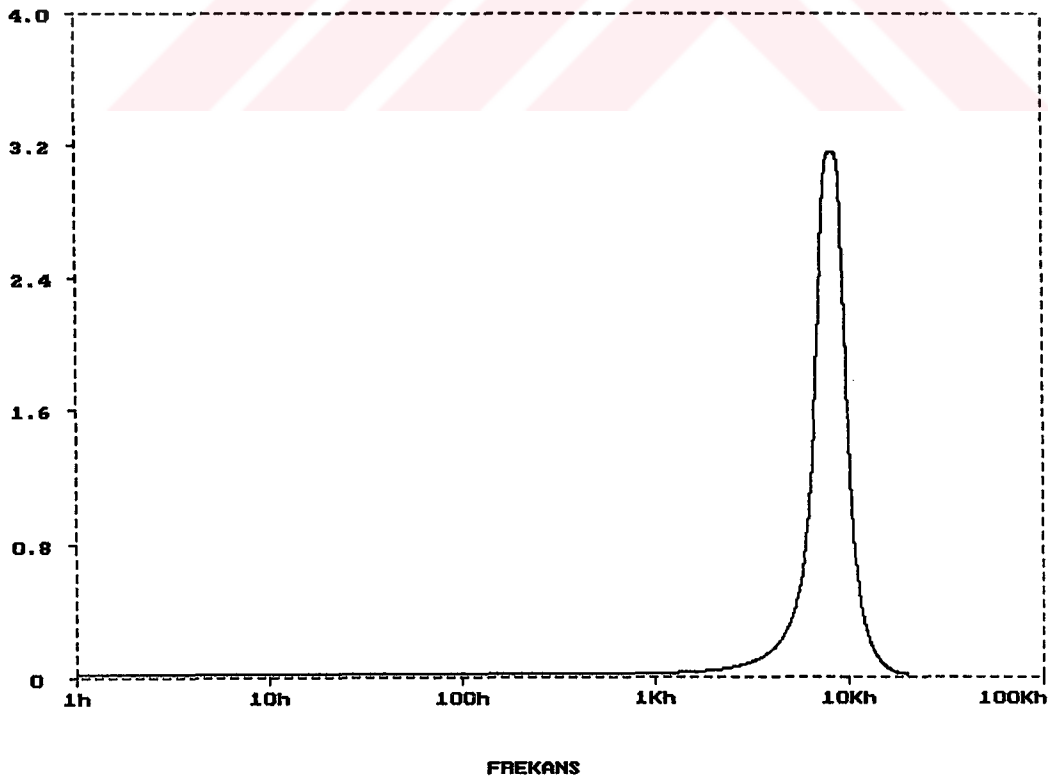
Şekil 5.6 Şekil 5.5’de görülen transfer fonksiyonuna ilişkin frekans eğrisi

Ana menüden band geçiren filtre seçeneği seçilirse yine Şekil 5.2'dekine benzer bir parametre menüsü ile karşılaşılır. Fakat bu menüdeki parametreler biraz daha farklıdır. Burada F_{cl} , alt kesim frekansını, F_{ch} , üst kesim frekansını, F_c , örnekleme frekansını, N , istenen fonksiyonun derecesini, K_v , dB olarak filtre kazancını, MKD, devrede yer alması istenen minimum kapasite değerini simgeler.

$$H(Z) = \frac{0.093409(1-Z^{-1})^2 (1+Z^{-1})^2}{1 - 0.885863Z^{-1} + 1.657540Z^{-2} - 0.668939Z^{-3} + 0.576565Z^{-4}}$$

$C[1][1]=$	4.92025pf	$C[2][1]=$	5.25100pf
$C[1][2]=$	4.92025pf	$C[2][2]=$	5.25100pf
$C[1][3]=$	5.30159pf	$C[2][3]=$	4.68688pf
$C[1][4]=$	4.00000pf	$C[2][4]=$	4.00000pf
$C[1][5]=$	18.80398pf	$C[2][5]=$	15.90558pf
$C[1][6]=$	12.09878pf	$C[2][6]=$	13.18096pf
$C[1][7]=$	4.00000pf	$C[2][7]=$	4.00000pf

Şekil 5.7 Program tarafından hesaplanan band geçiren filtre fonksiyonu ve bu fonksiyona ilişkin kapasite değerleri



Şekil 5.8 Şekil 5.7'de görülen transfer fonksiyonuna ilişkin frekans eğrisi

Örneğin $F_{cl}=7142.645996\text{Hz}$, $F_{ch}=9623\text{Hz}$, $F_c=40000\text{Hz}$, $N=4$, $K_v=10$, $MKD=4\text{pf}$ olarak girilirse Şekil 5.7’de görülen transfer fonksiyonu ve devrede yer alan kapasite değerleri elde edilir. Bu transfer fonksiyonun program tarafından çizilen transfer eğrisi de Şekil 5.8’de görülmektedir.



SONUÇ VE ÖNERİLER

Bu tez çalışmasında, ikinci kuşak akım taşıyıcı ve anahtarlı kapasiteler kullanarak genel ikinci dereceden gerilim transfer fonksiyonlarını gerçekleyen SC aktif filtre devre yapısı tasarlanmıştır. Bu yapı özelleştirilerek alçak geçiren, yüksek geçiren, band geçiren ve notch tipi filtre yapıları oluşturulmuştur. Ayrıca integratör ve türev alıcı devreleri de elde edilmiştir. Tasarım metodu, işaret akış diyagramından direkt olarak devre modeline geçişi sağlamaktadır. Böylece eleman değerleri, transfer fonksiyonun katsayılarına göre elde edilebilmektedir. Bu yöntem sayesinde tasarım eşitlikleri basitleşmektedir. Kararlılık incelemesi yapıldığında devrenin kompleks kararlı kutupları ve aynı işaretli reel kutupları gerçekleyebildiği fakat ters işaretli reel kararlı kutupları gerçekleyemediği görülmüştür. Bunun sebebi paydada yer alan z^{-2} teriminin katsayısı olan b_2 'nin hiçbir zaman negatif olamamasıdır. Ayrıca devrenin genlik fonksiyonu ve katsayı duyarlılığı da incelenmiştir. Devrede yer alan tüm kapasitelerin aynı anda değişmesi ve bağıl değişimlerinin aynı olması durumunda genlik transfer fonksiyonunun ve katsayıların bu değişimden etkilenmediği görülmüştür. Paydada yer alan katsayılar, bu değişimden etkilenmedikleri için kutuplar da bu değişimden etkilenmezler. Bir başka deyişle tüm elemanlarda meydana gelen değişimler birbirini kompanze etmekte ve böylece kutupların kararlı bölgeden kararsız bölgeye geçmesi ihtimali ortadan kalkmaktadır.

Devre topolojisinde gecikmeli iletkenlik adı verilen anahtarlı kapasite yapısında, kazancı bir olan bağımlı kaynak görülmektedir. Bu bağımlı kaynak, akım taşıyıcı eleman kullanılarak oluşturulan bir buffer devresi ile gerçekleştirilmiştir. Genel filtre topolojisinde H kapasitesinin bağlı olduğu bağımlı kaynağa bir başka deyişle buffer'a gerek yoktur. Bunun yerine H kapasitesi, akım taşıyıcının y ucuna bağlanabilir. Bu yapıldığında örnekleme devresinin çıkışına buffer konması gerekir. Böylece giriş gelecek örneklenmiş veri domeninde bir bozulma söz konusu olmayacaktır.

Elde edilen bu yapılar, yukarıda belirtilen koşullar uyarınca spice simülasyon programı yardımıyla analiz edilmiştir. Analiz esnasında kullanılan yapılar ideal akım taşıyıcı ve ideal anahtar içermektedir. Devrelerin frekans karakteristikleri,

tasarlanmak istenen filtre transfer fonksiyonlarının karakteristiklerine tıpatıp uymaktadır. Ayrıca devrelerin girişine örneklenmiş veri işaretleri uygulanarak çıkış karakteristikleri incelenmiştir. Sonuçlar beklendiği gibi elde edilmiştir ve eklerde sunulmuştur.

Spice simülasyon programı ile yapılan inceleme sonucu $z = -1$ 'de sıfırı olan ve olmayan band geçiren filtre devrelerinin frekans ve darbe cevabı karakteristiklerinin hemen hemen aynı olduğu görülmüştür. $z = -1$ 'de sıfırı olmayan band geçiren filtre devresindeki toplam kapasite miktarı ve kullanılan kapasite sayısı diğerine göre daha düşüktür. Bu nedenlerden ötürü $z = -1$ 'de sıfırı olmayan band geçiren filtre devresi diğerine göre daha avantajlıdır.

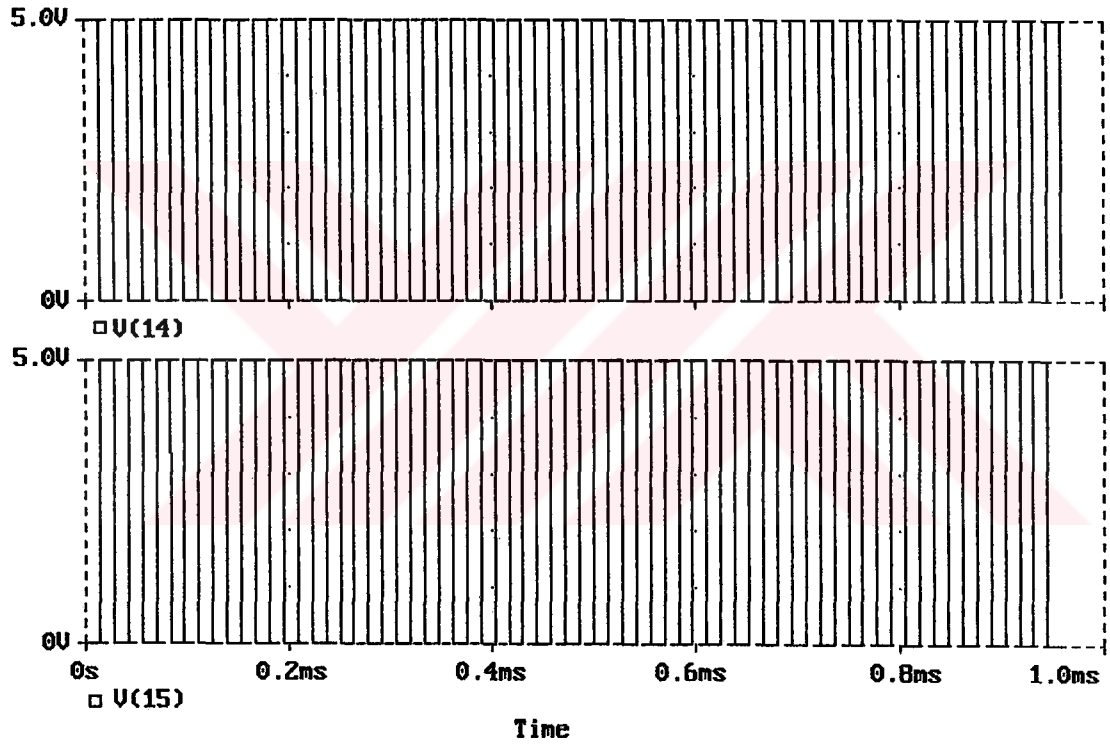
Akım taşıyıcı SC aktif filtrelerin ve z domeni filtre gerilim transfer fonksiyonlarının tasarım ilkeleri bilgisayar ortamına aktarılarak akım taşıyıcı içeren SC aktif filtrelerin tasarımını yapan C dilinde bir bilgisayar programı hazırlanmıştır. Hazırlanan program devreye ait normalize kapasite değerlerini hesaplamakta ve fonksiyona ilişkin frekans karakteristiğini çizmektedir. Çizilen bu karakteristikler, spice simülasyon programı ile elde edilen frekans karakteristikleri ile kıyaslanmış ve hemen hemen birbirlerinin aynısı olduğu görülmüştür.

KAYNAKLAR

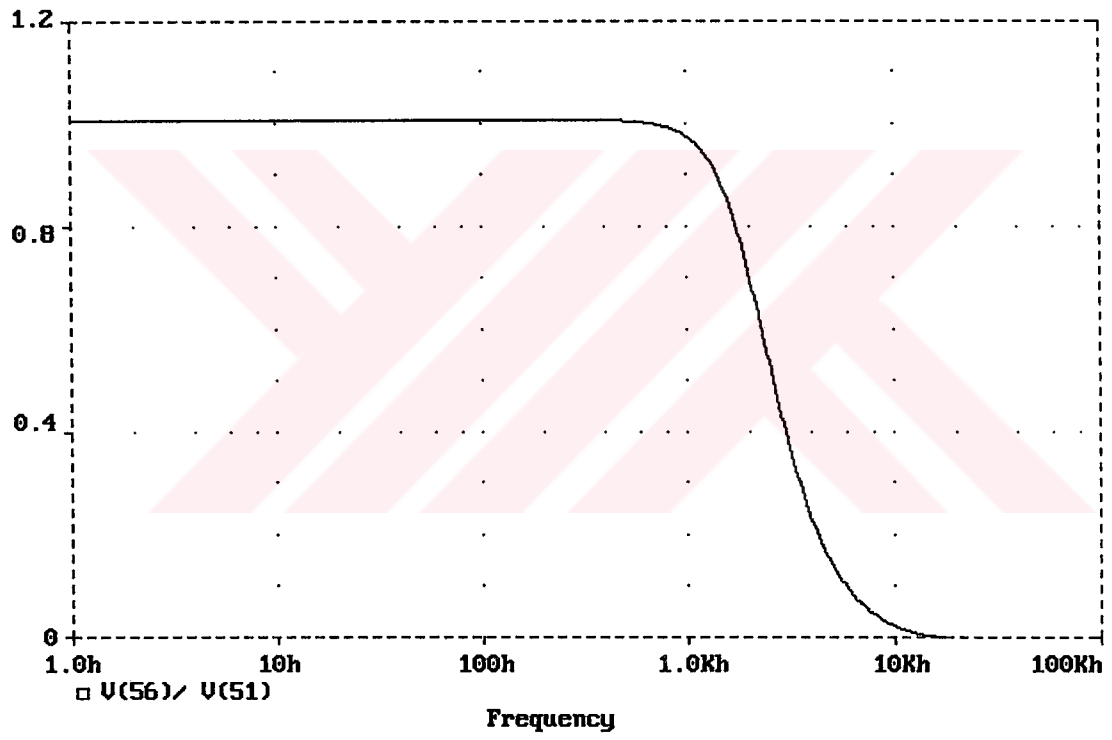
- [1] **CAVES, J. T., COPELAND, M. A., RAHİM, C. F. and ROSENBAUM, S. D.** "Sampled Analog Filtering Using Switching Capacitor as Resistor Equivalents ", IEEE Journal Solid-State Circuitc, Vol: SC-12, No. 6, December 1977
- [2] **KURTH, C. F.** " Nodal Analysis of Switched-Capacitor Network ", IEEE Transactions on Circuit and System, Vol. Cas-26, No. 2, February 1979.
- [3] **KURTH, C. F. and MOSCHYTZ, G. S.** "Two-Port Analysis of Switched-Capacitor Networks Using Four-Port Equivalent Circuits in the z-Domain ", IEEE Transactions on Circuit and System, Vol. Cas-26, No. 3, March 1979.
- [4] **LAKER, K. R.** " Equivalent Circuits for the Analysis and Synthesis of Switched Capacitor Network ", Bell Syst. Tech. J. 58, 729-769 (1979).
- [5] **SZENTIRMAI, G. and TEMES, G. C.** " Switched-Capacitor Building Blocks ", IEEE Transactions on Circuit and System, Vol. Cas-27, No. 6, June 1980.
- [6] **ZHOU, Z.** " A Simplified Design-Oriented Analysis of Switched-Capacitor Active Filters ", Circuit Theory and Applications, Vol. 12 179-189 (1984).
- [7] **ANDAY, F.** " Synthesis of Switched- Capacitor Active Filters : z-Domain Equivalent Admittance Approach ", Circuit Theory and Application, Vol. 15. 85-87 (1987).
- [8] **ONO, T.** "Switched Capacitor Pseudo Current Conveyor", Circuit Theory and Design, 717-722 (1993).
- [9] **MAUNDY, B. J. and ARONHIME, B. P.** "Switched-Capacitor Current Conveyor Building Blocks", Proc. of ISCAS, 1996
- [10] **GHAUSI, M. S. and LAKER, K. R.** "Modern Filter Design : Active Switched Capacitor Sampled-Data Networks ", Prentice Hall, Inc. , 1981, (376-500).
- [11] **ACAR, C.** " Elektrik Devrelerinin Analizi : Çok Uçlu ve Çok Kapılı Devre Elemanları", İstanbul Teknik Üniversitesi, 1995, (290-302).
- [12] **ANDAY, F. and GÜNEŞ, E. O.** " Realization of nth-Order Transfer Functions Using Current Conveyors ", International Journal of Circuit Theory and Applications, Vol. 20, 693-696 (1992).
- [13] **RASHID, M.** " SPICE for Circuits and Electronics Using Pspice ", Prentice Hall, Inc., 1990.

EK-A

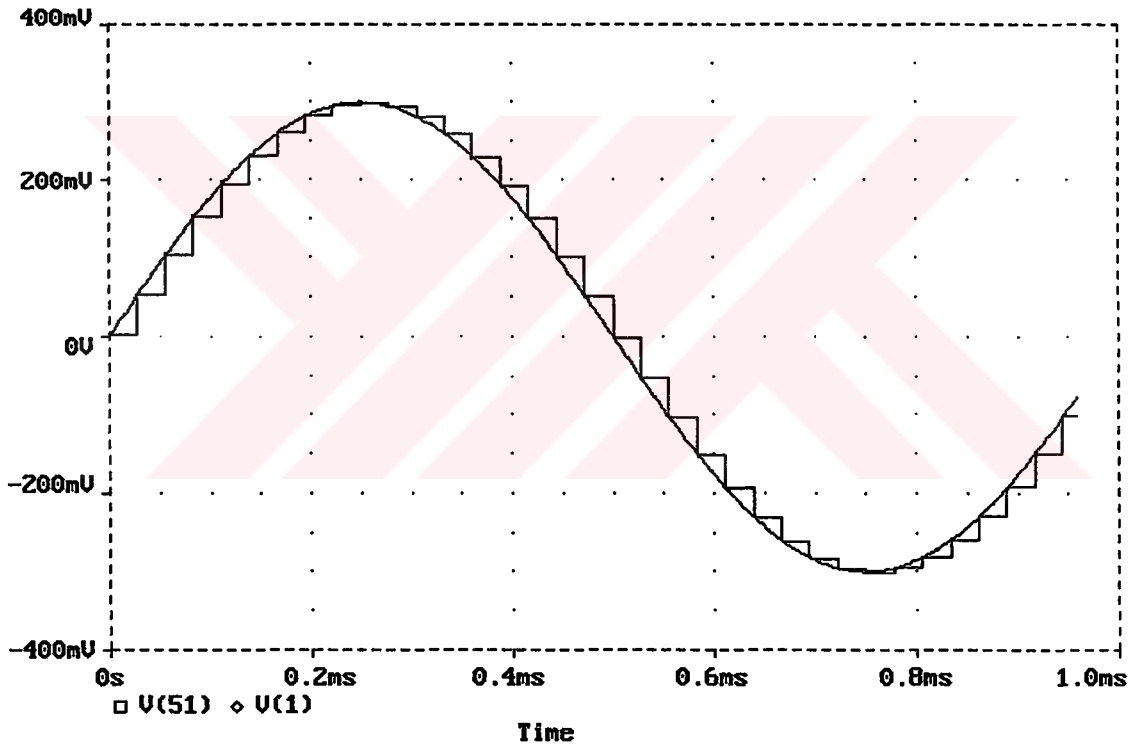
ALÇAK GEÇİREN DEVRENİN SPICE ÇIKIŞ DOSYASI



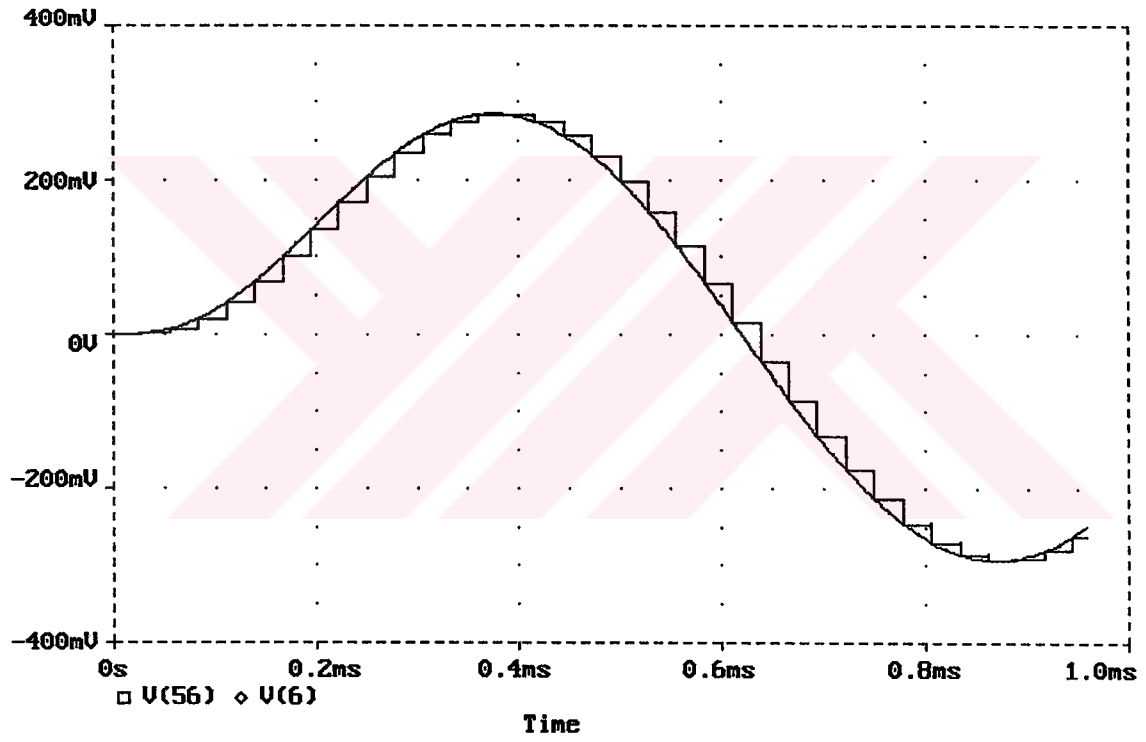
Şekil A.1 Devreye ilişkin çift ve tek saat darbeleri



Şekil A.2 Frekans karakteristiği



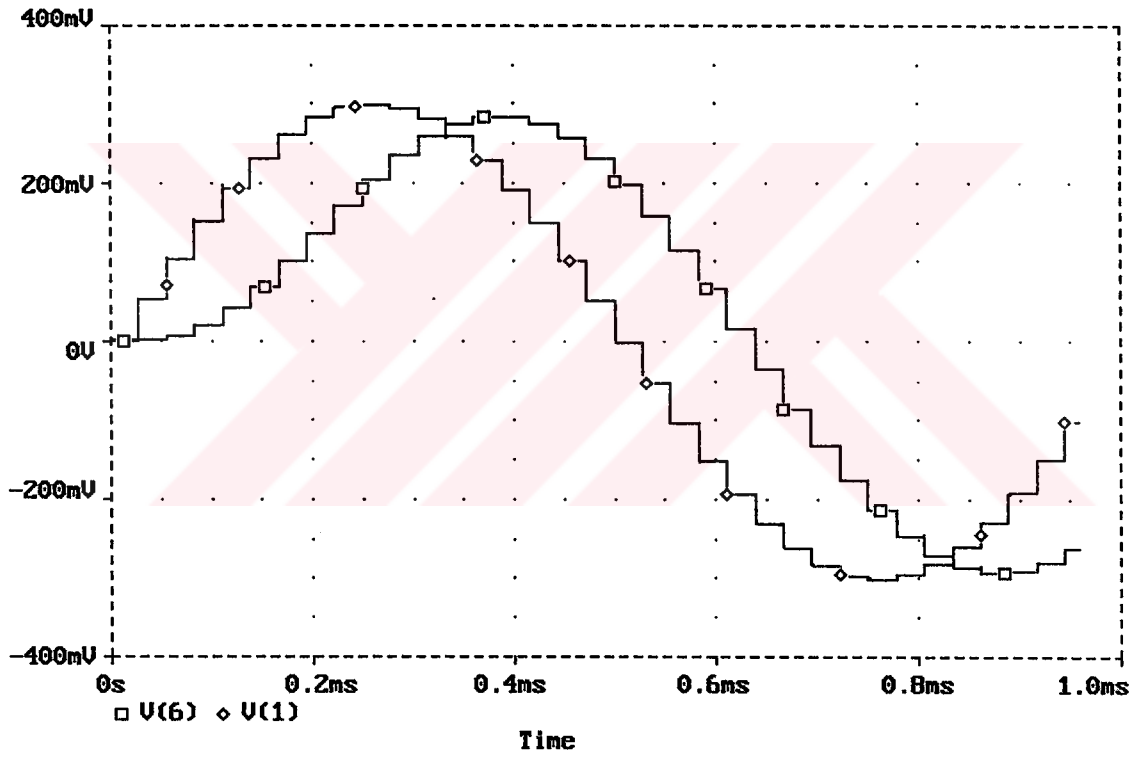
Şekil A.3 Devrenin girişine uygulanan işaret
 V(51) : Örnekleme devresi girişi
 V(1) : Filtre girişine uygulanan, 36 kHz' de örneklenmiş giriş işareti



Şekil A.4 Devrenin çıkışında elde edilen işaretler

V(56) : Sinüzoidal eşdeğer devresinin çıkışında elde edilen işaret

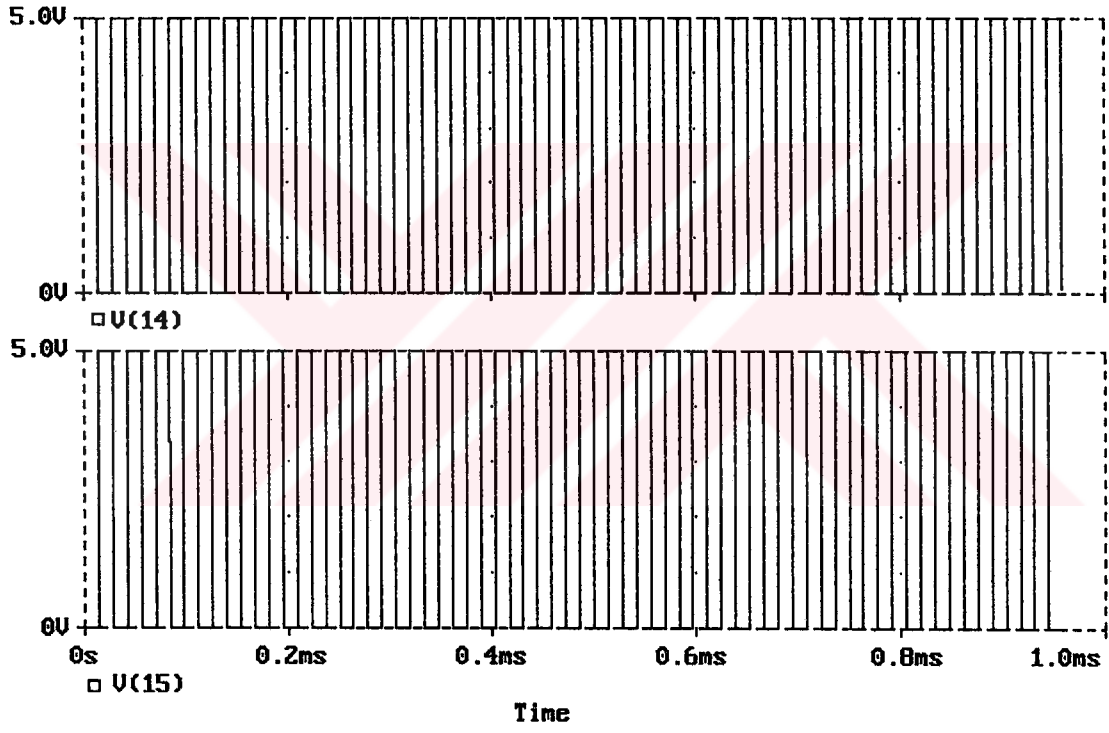
V(6) : Filtre çıkışında elde edilen işaret



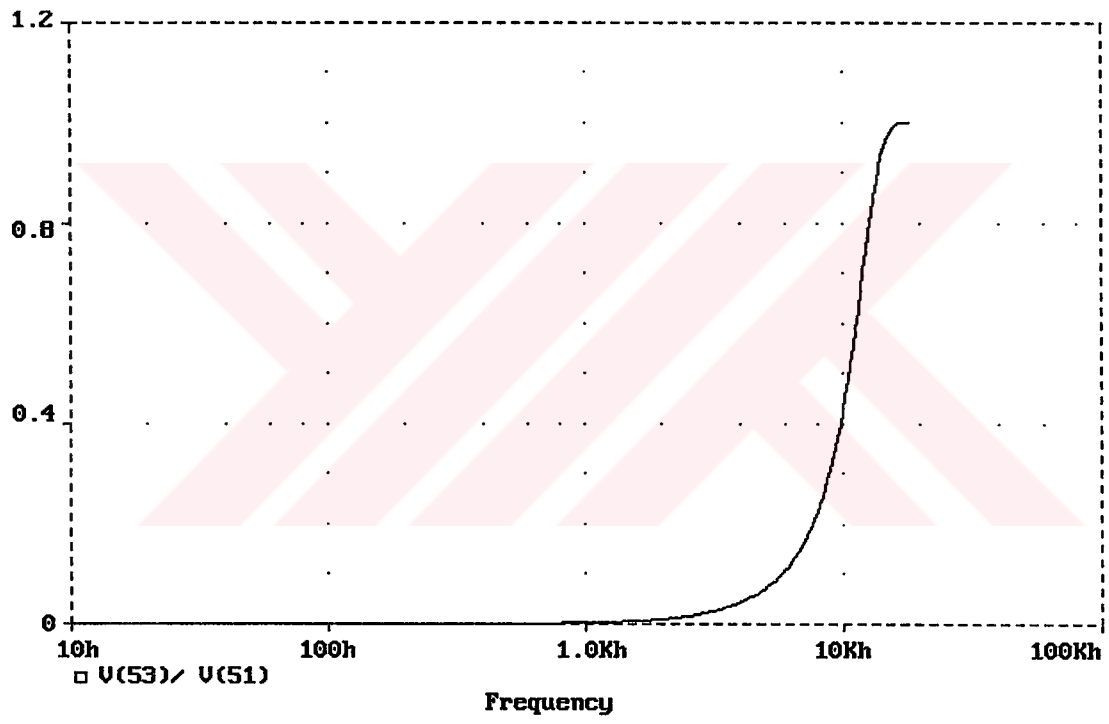
Şekil A.5 Örnekleilmiş giriş ve çıkış karakteristikleri
 $V(1)$: Filtre girişine uygulanan giriş işareti
 $V(6)$: Filtre çıkışında elde edilen çıkış işareti

EK-B

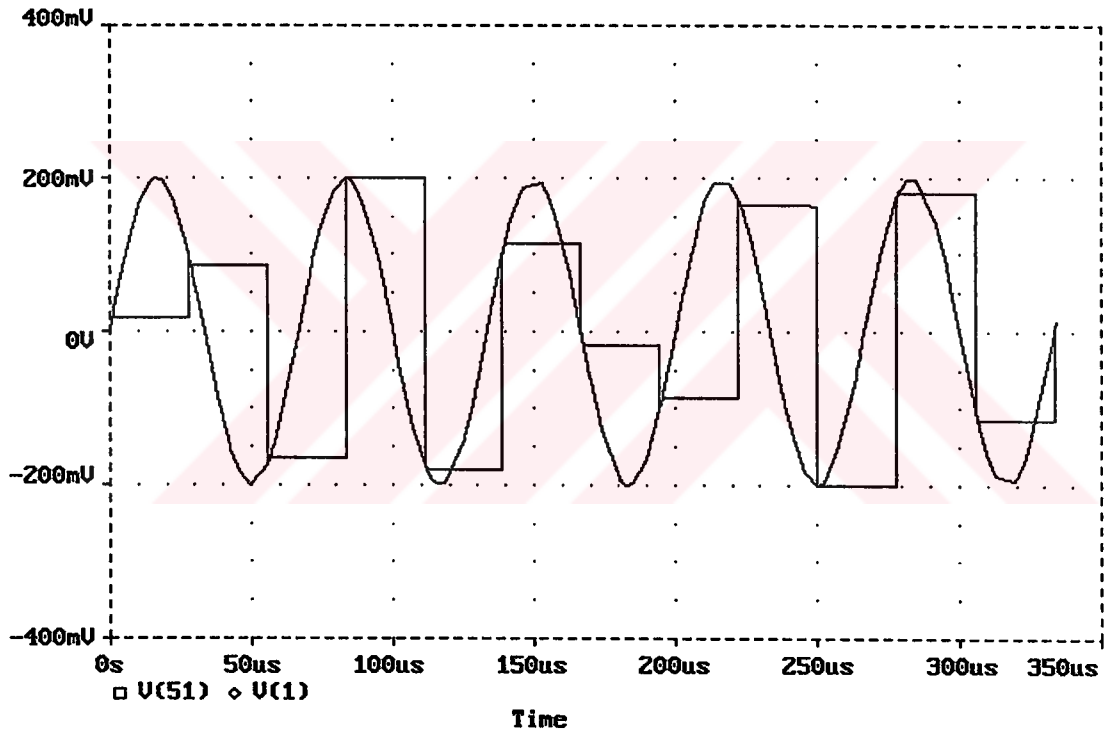
YÜKEK GEÇİREN DEVRENİN SPICE ÇIKIŞ DOSYASI



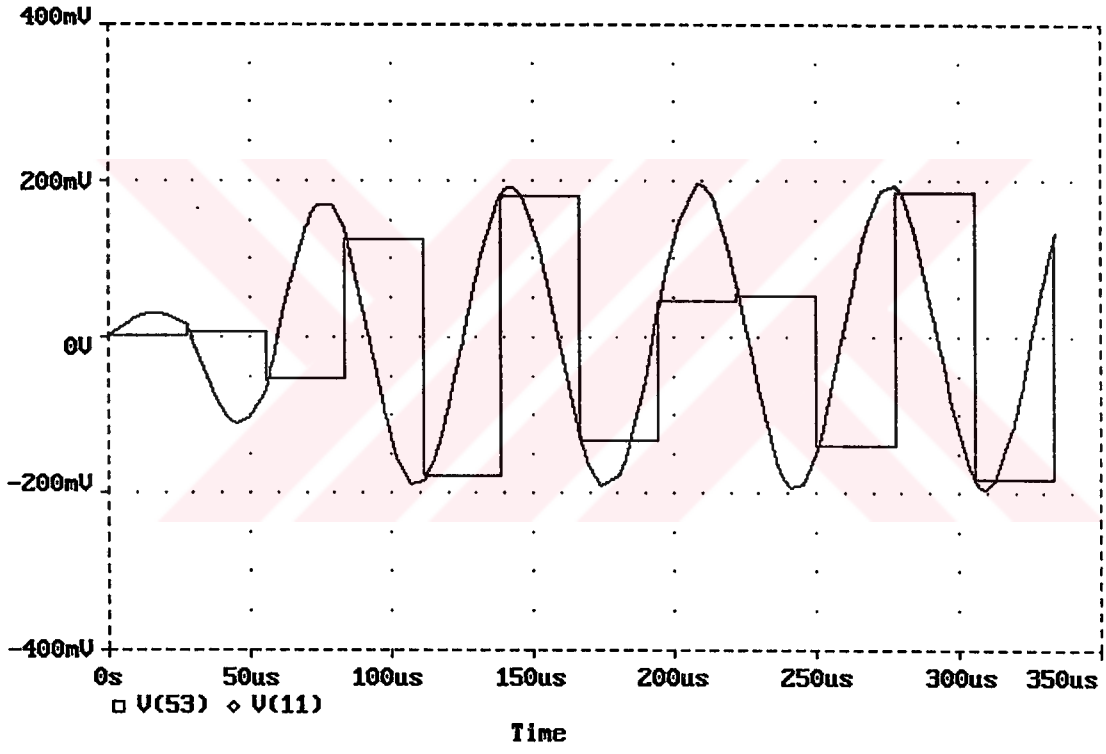
Şekil B.1 Devreye ilişkin çift ve tek saat darbeleri



Şekil B.2 Frekans karakteristiği



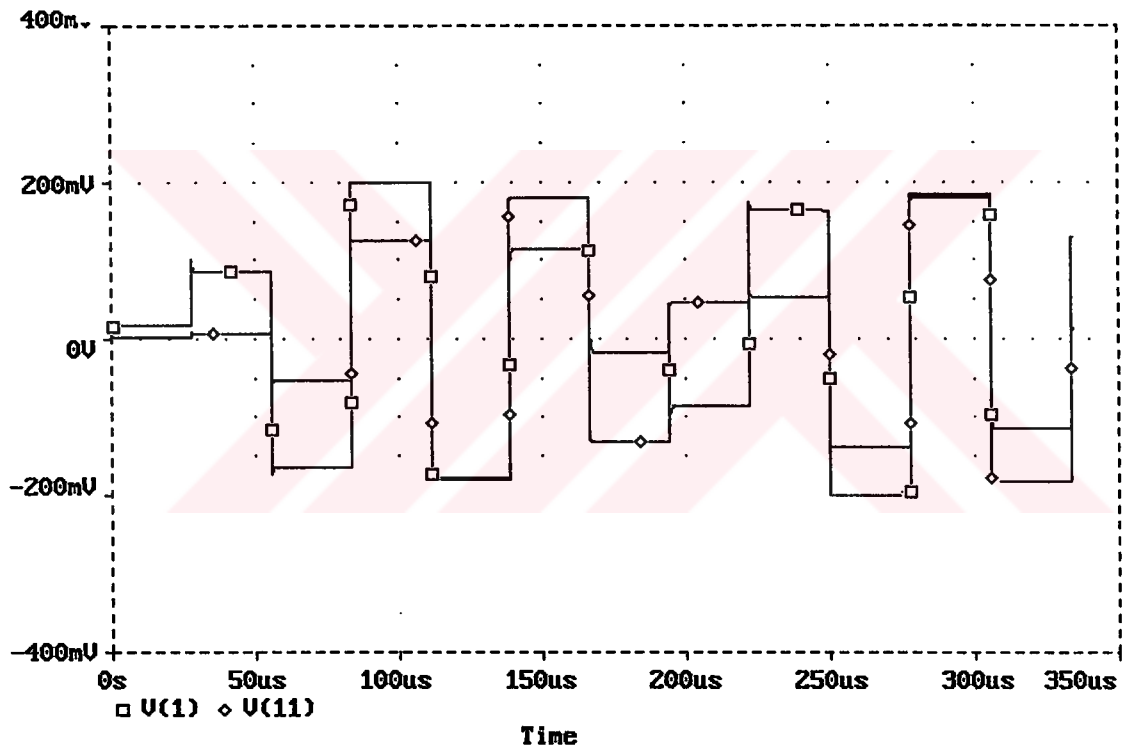
Şekil B.3 Devrenin girişine uygulanan işaret
 V(51) : Örnekleme devresi girişi
 V(1) : Filtre girişine uygulanan, 36 kHz' de örneklenmiş giriş işareti



Şekil B.4 Devrenin çıkışında elde edilen işaretler

V(53) : Sinüzoidal eşdeğer devresinin çıkışında elde edilen işaret

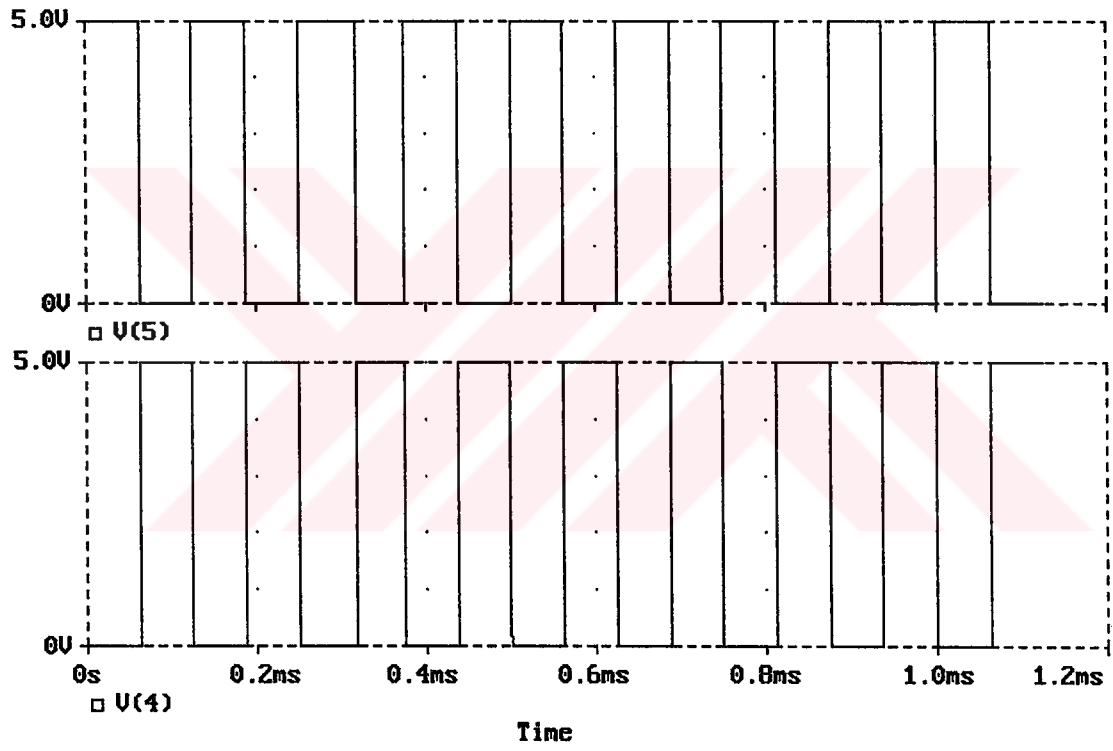
V(11) : Filtre çıkışında elde edilen işaret



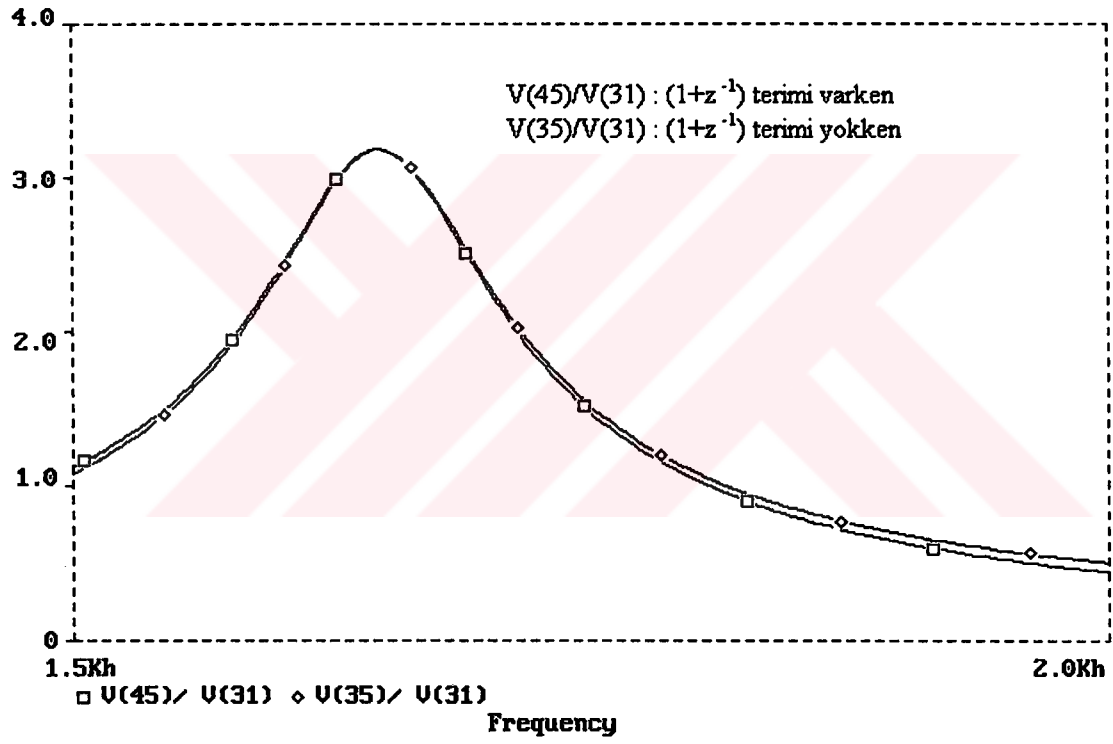
Şekil B.5 Örnekleilmiş giriş ve çıkış karakteristikleri

EK-C

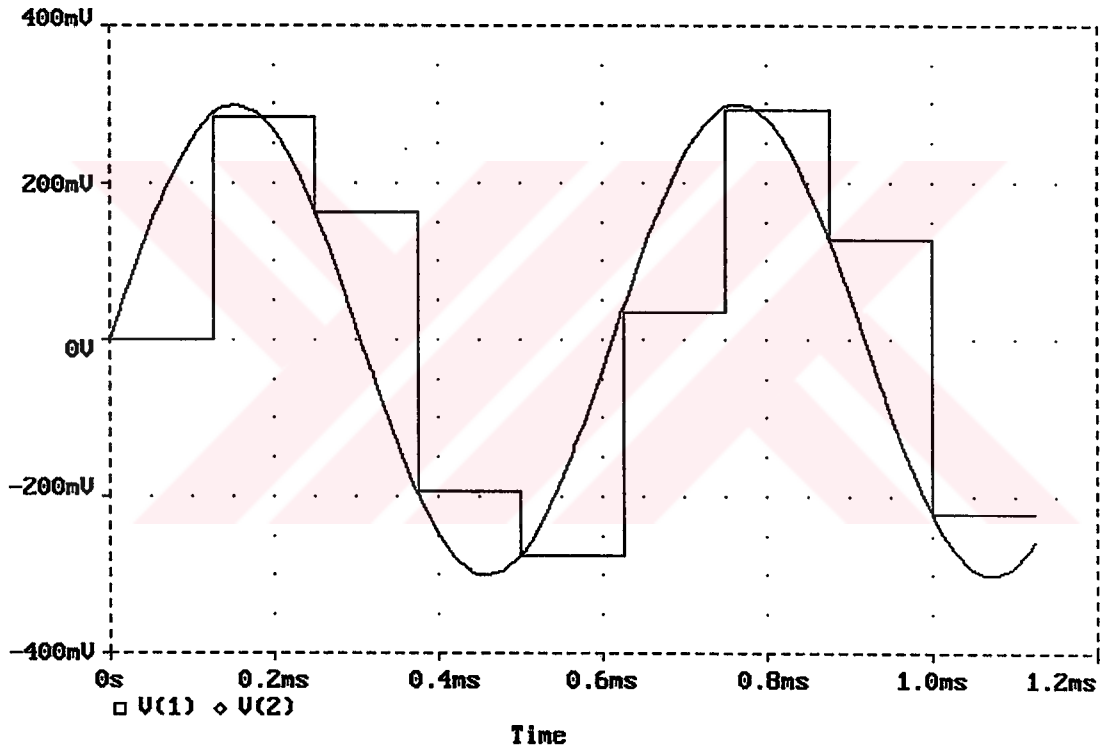
BAND GEÇİREN DEVRENİN SPICE ÇIKIŞ DOSYASI



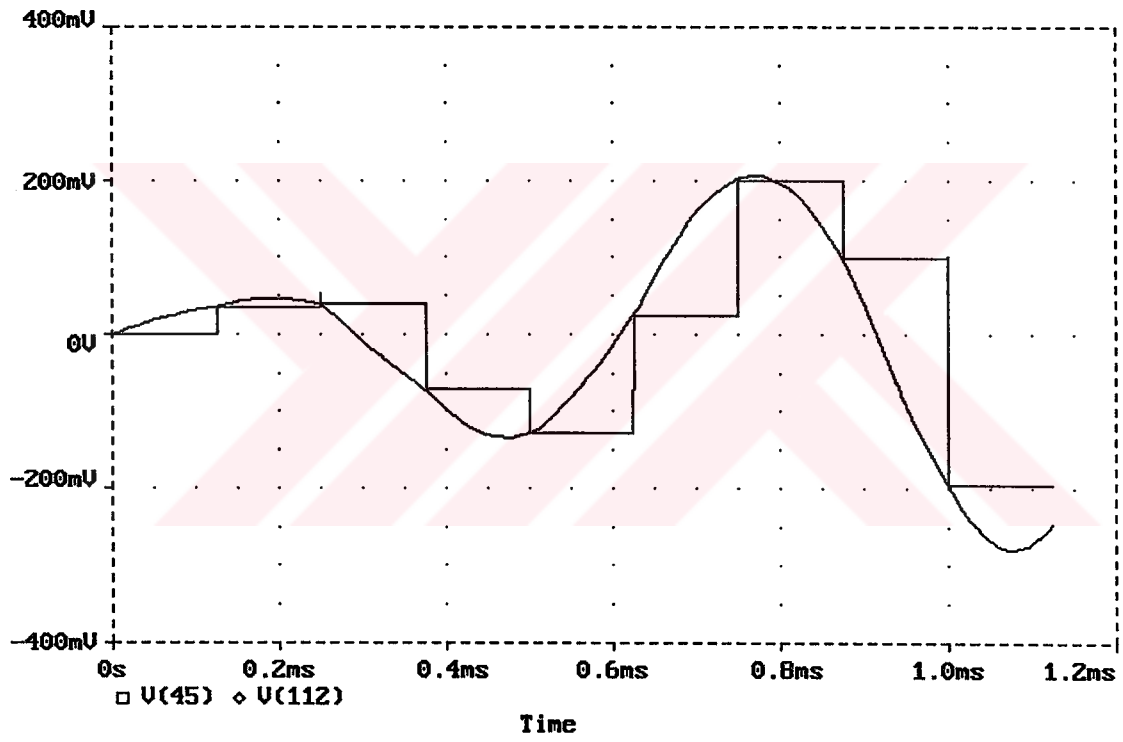
Şekil C.1 Devreye ilişkin çift ve tek saat darbeleri



Şekil C.2 Frekans karakteristiği



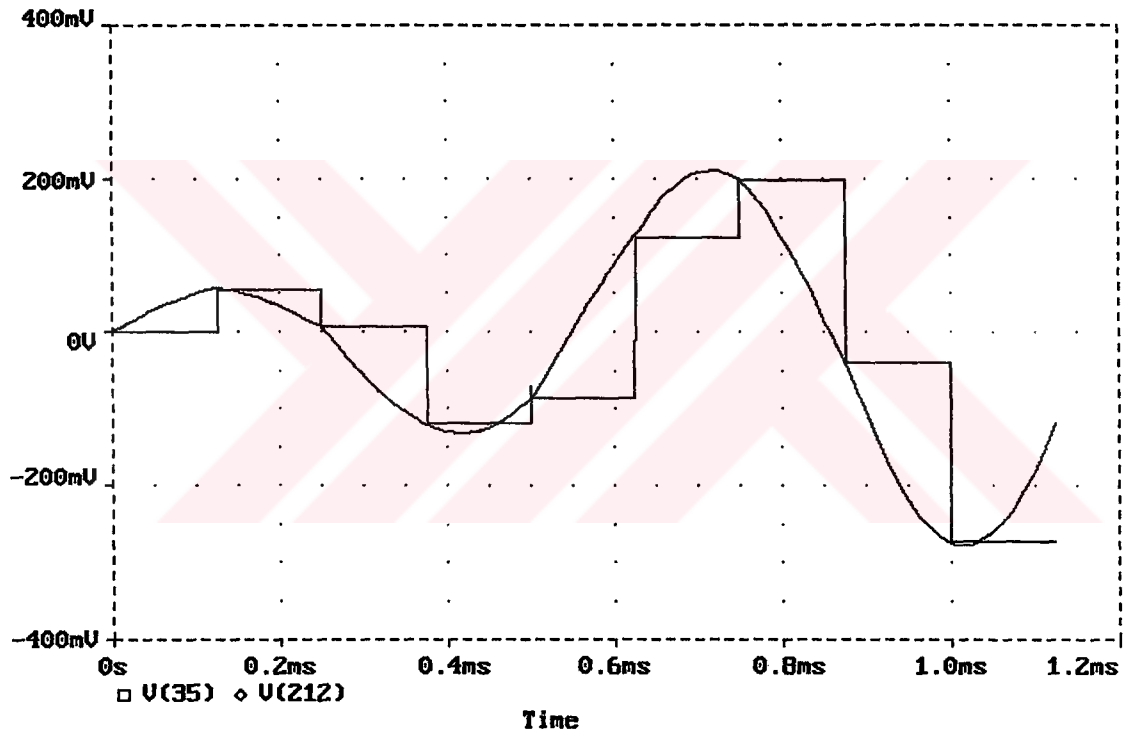
Şekil C.3 Devrenin girişine uygulanan işaret
 V(2) : Örnekleme devresi girişi
 V(1) : Filtre girişine uygulanan, 8 kHz' de örneklenmiş giriş işareti



Şekil C.4 $(1+z^{-1})$ terimine sahip olan devrenin çıkışında elde edilen işaretler

V(45) : Sinüzoidal eşdeğer devresinin çıkışında elde edilen işaret

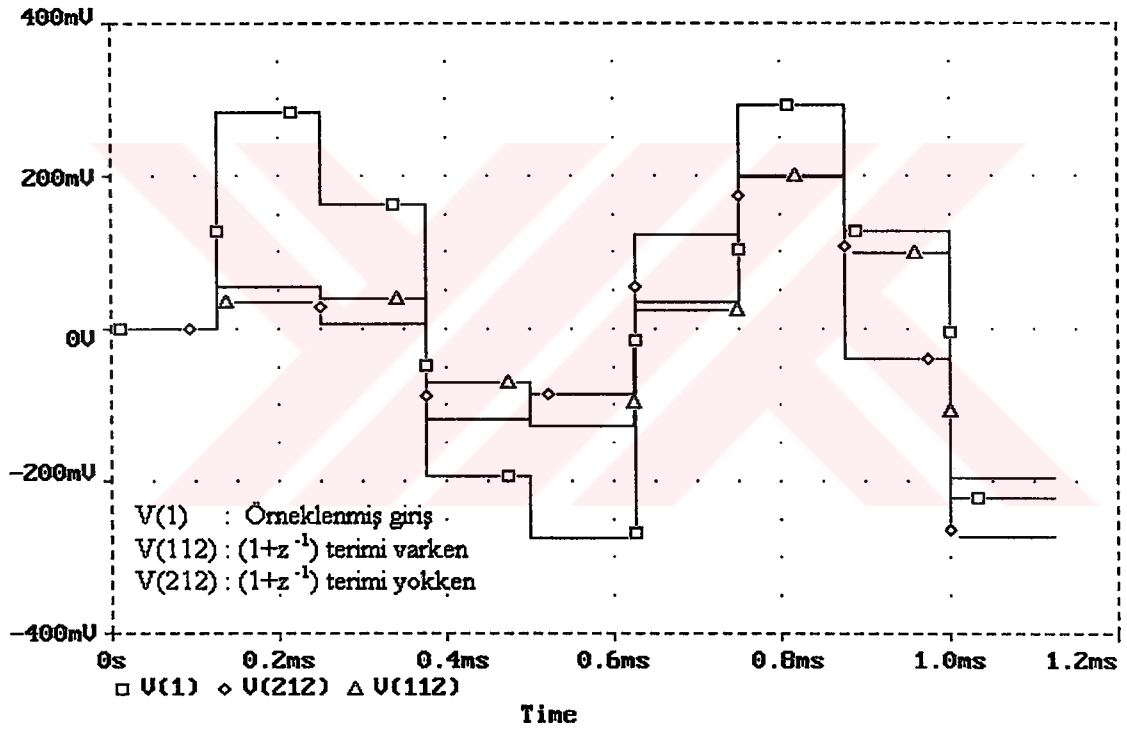
V(112) : Filtre çıkışında elde edilen işaret



Şekil C.5 $(1+z^{-1})$ terimine sahip olmayan devrenin çıkışında elde edilen işaretler

V(35) : Sinüzoidal eşdeğer devresinin çıkışında elde edilen işaret

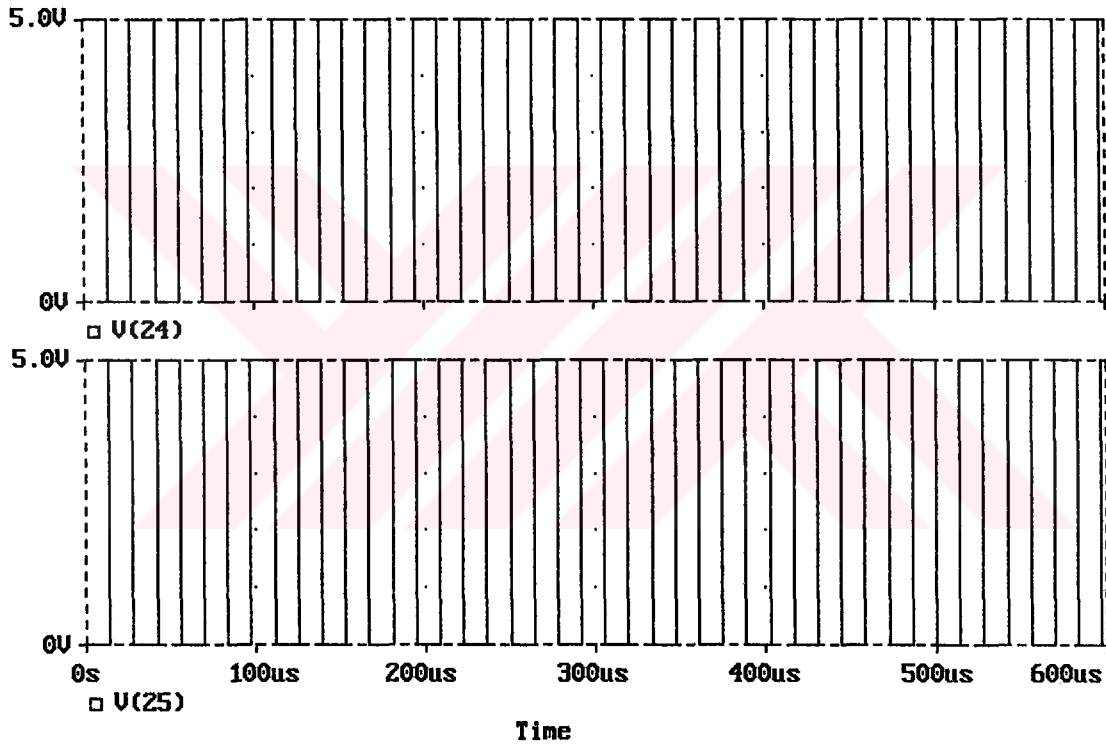
V(212) : Filtre çıkışında elde edilen işaret



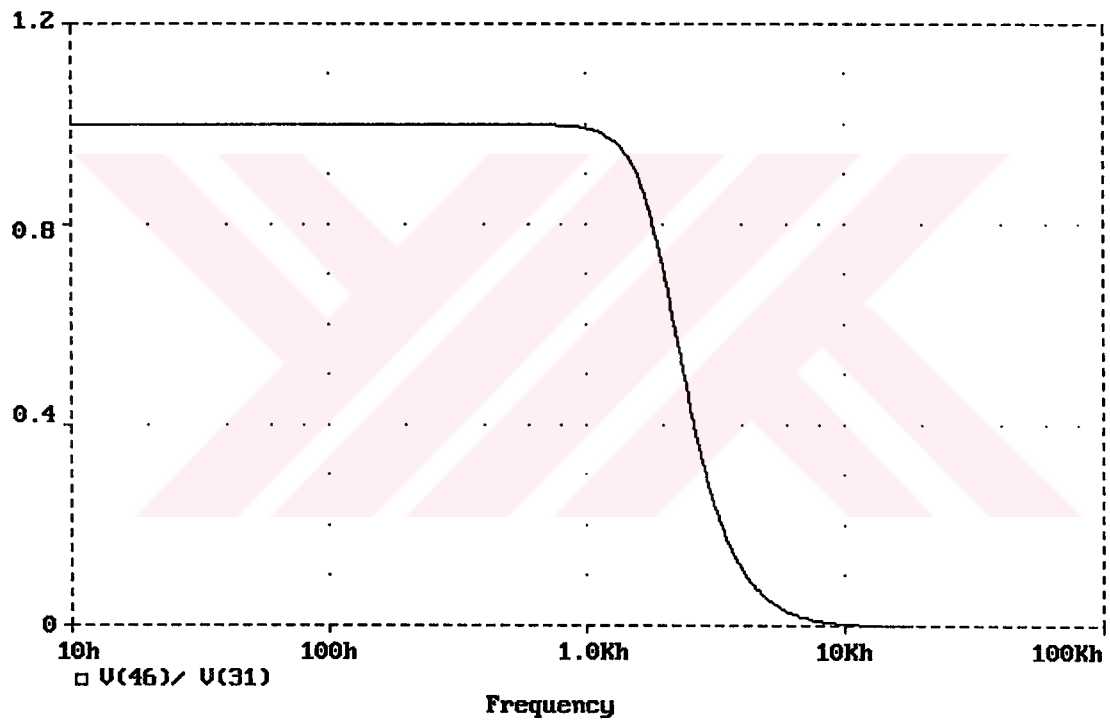
Şekil C.6 Örnekleme giriş ve çıkış karakteristikleri

EK D

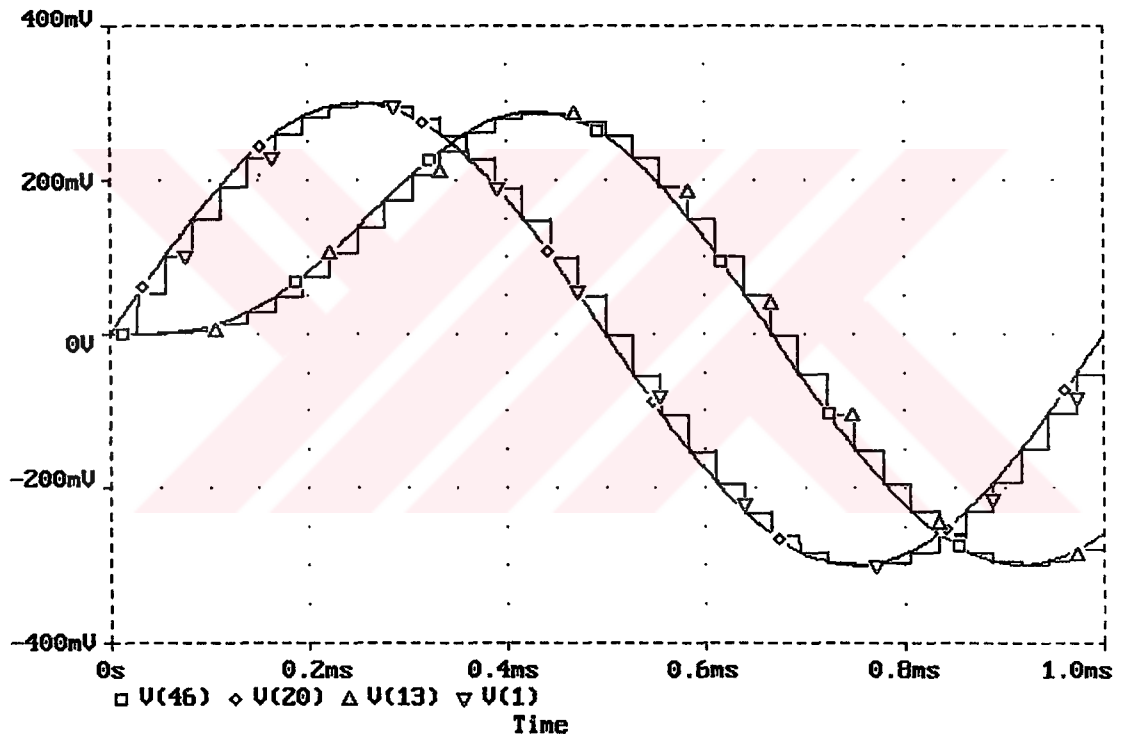
3.DERECEDEN ALÇAK GEÇİREN FİLTRENİN SPICE ÇIKIŞ DOSYASI



Şekil D.1 Devreye ilişkin çift ve tek saat darbeleri



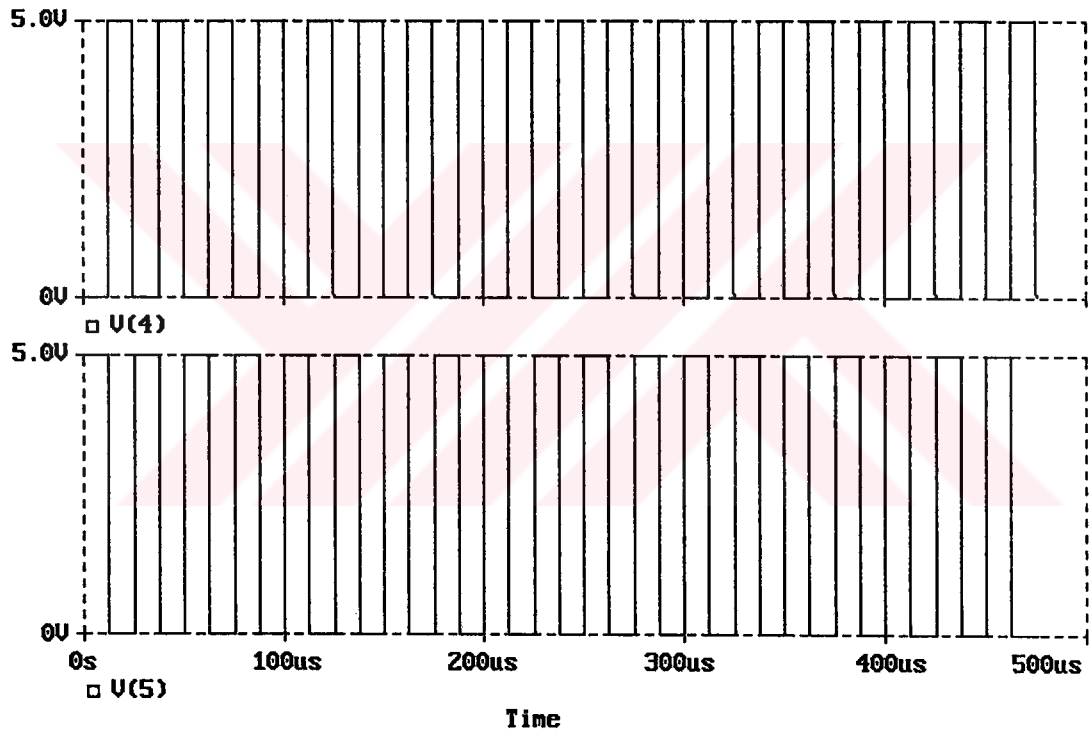
Şekil D.2 Frekans karakteristiği



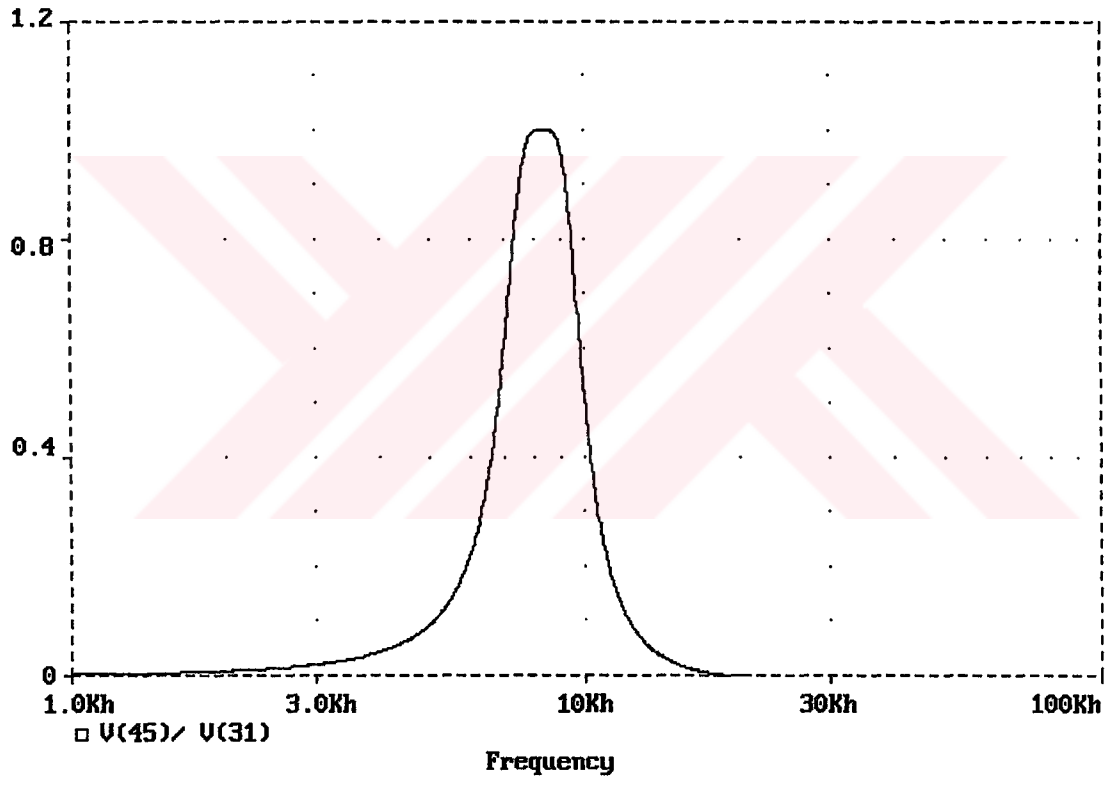
Şekil D.3 Örneklenmiş giriş ve çıkış karakteristikleri
 V(20) : Örnekleme devresi girişi
 V(1) : Filtre girişine uygulanan, 36kHz'de örneklenen giriş işareti
 V(13) : Filtre çıkışında elde edilen çıkış işareti
 V(46) : Sinüzoidal eşdeğer devrenin çıkışında elde edilen sinüs işareti

EK E

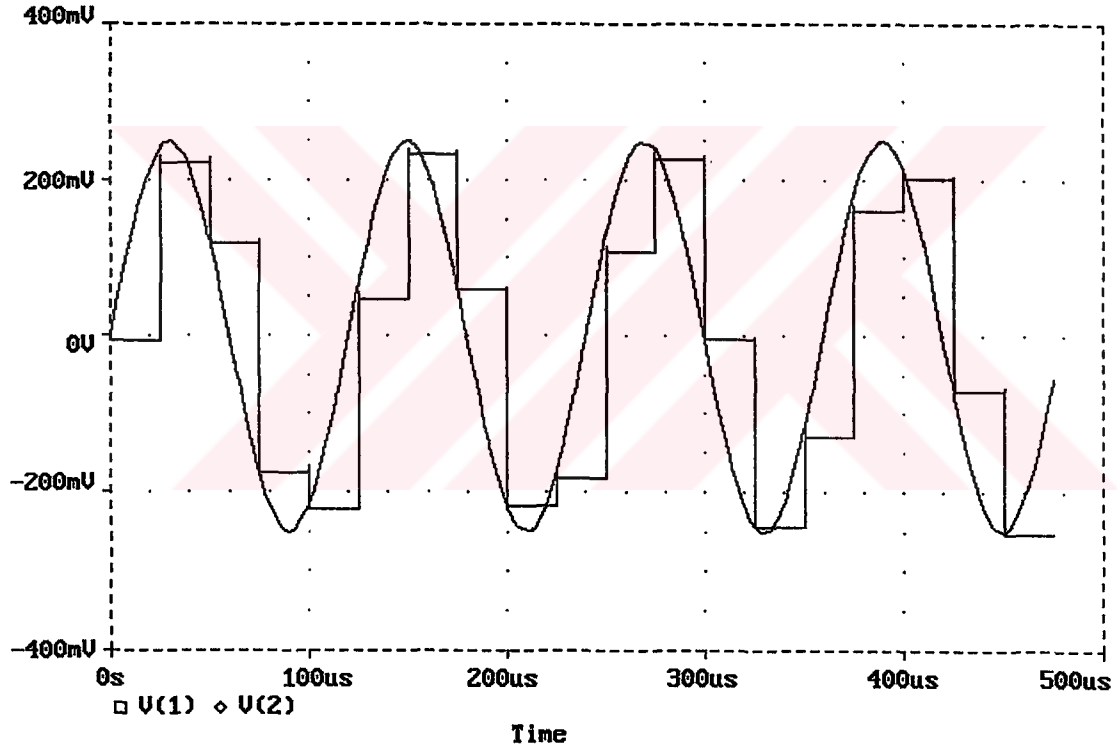
4.DERECEDEN BAND GEÇİREN FİLTRENİN SPICE ÇIKIŞ DOSYASI



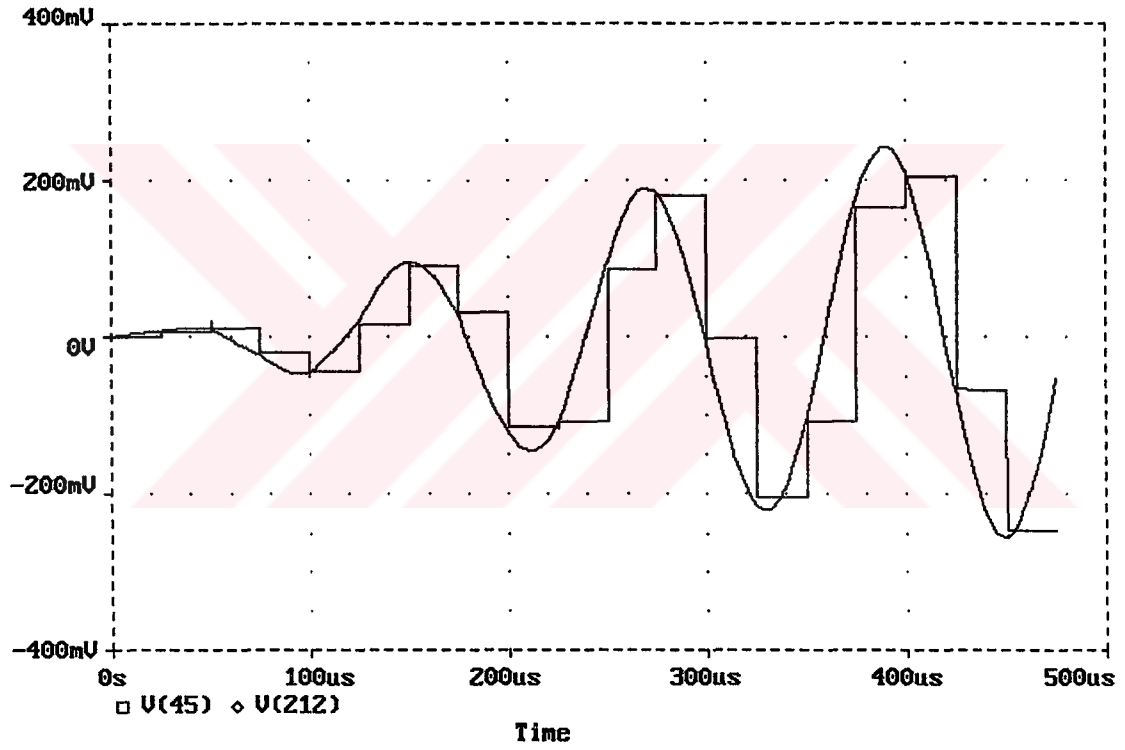
Şekil E.1 Devreye ilişkin çift ve tek saat darbeleri



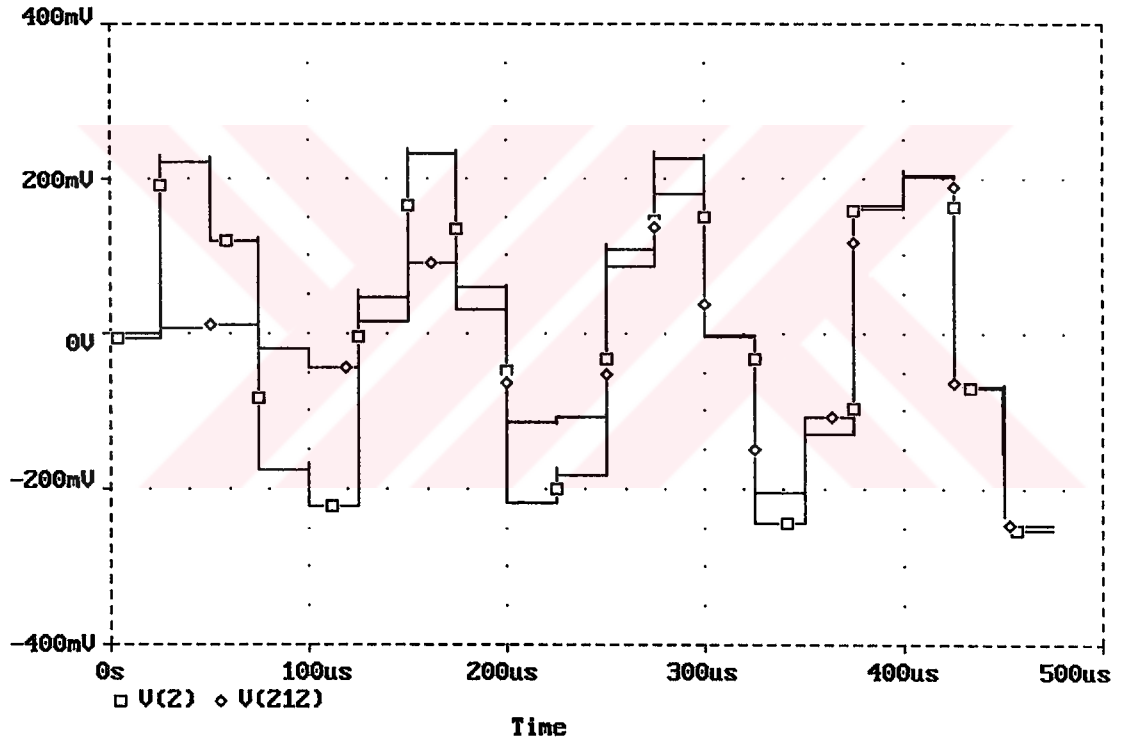
Şekil E.2 Frekans karakteristiği



Şekil E.3 Devrenin girişine uygulanan işaret
 V(2) : Örnekleme devresi girişi
 V(1) : Filtre girişine uygulanan, 40 kHz' de örneklenmiş giriş işareti



Şekil E.4 Devrenin çıkışında elde edilen işaretler
 V(45) : Sinüzoidal eşdeğer devresinin çıkışında elde edilen işaret
 V(212) : Filtre çıkışında elde edilen işaret



Şekil E.5 Örneklenmiş giriş ve çıkış karakteristikleri

EK-F

BİLGİSAYAR PROGRAMI

FİLTRE.CPP

```
#include<stdio.h>
#include<graphics.h>
#include<process.h>
#include<complex.h>
#include<math.h>
#include<stdlib.h>
#include<dos.h>
#include<bios.h>
#include<conio.h>
#include<fcntl.h>
#include<string.h>
#include<sys\stat.h>
#include "cursor.h"
#define PI (float) (3.141592654)
#define CR 13
#define ESC 27
#define BACK_SPACE 8
#define HOME 327
#define END 335
#define YUKARI_TUSU 328
#define ASAGI_TUSU 336
#define SOL_TUSU 331
#define SAG_TUSU 333
#define PAGE_UP 329
#define PAGE_DOWN 337
#define F1 315 /*AGS*/
#define F2 316 /*HGS*/
#define F3 317 /*BGS*/
#define F4 318 /*YARDIM*/
#define F5 319 /*ÇIKIŞ*/
#define F6 320
#define F7 321 /*YÜKLEME*/
#define F8 322 /*SAKLAMA*/
#define F9 323 /*HESAPLAMA*/
#define F10 324 /*ANA MENÜYE DÖNÜŞ*/
#define F11 325
#define F12 326
#define DEL 339 /*SATIR SİLME*/
FILE *dosya;
void ags();
menu();
void directory();
void grafik();
int KAR,sayfa,sayfa1,ara;
float eksenx_max,eksenx_max1,eksenx,ekseny[200],ekseny_max,fx[210],fy[210];
char BITTI1,BITTI11,BITTI2,BITTI3,BITTI4,BITTI5;
```

```

char METIN[200], isim[30], sakla[20000];
int satir;
fpos_t POZIS;
int u_x(float u)
{
    return((int)(log10(u)*570*(1/log10(eksenx_max1))));
}
int v_y(float v)
{
    return((int)(390+(v*325)*(120/((ceil(ekseny_max+0.2))*100))));
}
void UKS_line (float u1, float v1, float u2, float v2)
{
    line ( u_x(u1), v_y(v1), u_x(u2), v_y(v2));
}
void UKS_outtextxy(float u1, float v1)
{
    outtextxy(u_x(u1), v_y(v1), ".");
}
void UKS_circle (float u, float v, float r)
{
    circle ( u_x(u), v_y(v), u_x(r) );
}

void grafik()
{
    int GrAdapter, GrMod;
    int hata_kodu, bol, sayac1=0, sayac2;
    float ymax, boly;
    int ilkcurposx, ilkcurposy;
    int soncurposx, soncurposy;
    int boyut, bitti6=0, l, belirle=0;
    int ESCTusu, eksy[200], eksx[200];
    char katar[31];
    detectgraph ( &GrAdapter, &GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter, &GrMod, "" );
    hata_kodu = graphresult();
    if(hata_kodu != grOk)
    {
        printf("grafik hatası:(#%d kodlu hata)!s\n", hata_kodu, grapherrormsg(hata_kodu));
        getch();
        exit(1);
    }
    setlinestyle(3, 10, 1);
    line(60, 9, 637, 9);
    line(67, 9, 67, 401);
    line(60, 401, 637, 401);
    line(637, 9, 637, 401);
    line(60, 87, 67, 87);
    line(60, 166, 67, 166);
    line(60, 244, 67, 244);
    line(60, 323, 67, 323);
    line(60, 401, 67, 401);

    ymax=ceil(ekseny_max+0.2);
    boly=(float)ymax/5;
    sprintf(katar, "%.1f", ymax);
    outtextxy(30, 7, katar);

```

```

sprintf(katar, "%.1f", ymax-boly);
outtextxy(30,85,katar);
sprintf(katar, "%.1f", ymax-(2*boly));
outtextxy(30,164,katar);
sprintf(katar, "%.1f", ymax-(3*boly));
outtextxy(30,242,katar);
sprintf(katar, "%.1f", ymax-(4*boly));
outtextxy(30,321,katar);
outtextxy(40,399,"0");
setlinestyle(0,1,1);
settextjustify(1,2);
if(eksensex_max1==1000000){
    bol=570/6 ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,162,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol,401,257,405);
    outtextxy(67+2*bol,410,"100h");
    line(67+3*bol,401,352,405);
    outtextxy(67+3*bol,410,"1Kh");
    line(67+4*bol,401,447,405);
    outtextxy(67+4*bol,410,"10Kh");
    line(67+5*bol,401,542,405);
    outtextxy(67+5*bol,410,"100Kh");
    line(67+6*bol,401,637,405);
    outtextxy(49+6*bol,410,"1MGh");
}
if(eksensex_max1==100000){
    bol=570/5 ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,181,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol,401,295,405);
    outtextxy(67+2*bol,410,"100h");
    line(67+3*bol,401,409,405);
    outtextxy(67+3*bol,410,"1Kh");
    line(67+4*bol,401,523,405);
    outtextxy(67+4*bol,410,"10Kh");
    line(67+5*bol,401,637,405);
    outtextxy(49+5*bol,410,"100Kh");
}
else if(eksensex_max1==10000){
    bol=int(570/4) ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,67+bol,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol+1,401,67+2*bol+1,405);
    outtextxy(67+2*bol+1,410,"100h");
    line(67+3*bol+1,401,67+3*bol+1,405);
    outtextxy(67+3*bol+1,410,"1Kh");
    line(67+4*bol+2,401,67+4*bol+2,405);
    outtextxy(53+4*bol+2,410,"10Kh");
}
else if(eksensex_max1==1000){
    bol=int(570/6) ;
    line(67,401,67,405);

```

```

        outtextxy(67,410,"1h");
        line(67+91,401,67+91,405);
        outtextxy(67+91,410,"3h");
        line(67+2*bol,401,67+2*bol,405);
        outtextxy(67+2*bol,410,"10h");
        line(67+2*bol+91,401,67+2*bol+91,405);
        outtextxy(67+2*bol+91,410,"30h");
        line(67+4*bol,401,67+4*bol,405);
        outtextxy(67+4*bol,410,"100h");
        line(67+4*bol+91,401,67+4*bol+91,405);
        outtextxy(67+4*bol+91,410,"300h");
        line(67+6*bol,401,67+6*bol,405);
        outtextxy(57+6*bol,410,"1Kh");
    }
    else if(eksenx_max1==100){
        bol=int(570/4) ;
        line(67,401,67,405);
        outtextxy(67,410,"1h");
        line(67+136,401,67+136,405);
        outtextxy(67+136,410,"3h");
        line(67+2*bol+1,401,67+2*bol+1,405);
        outtextxy(67+2*bol+1,410,"10h");
        line(67+2*bol+136+1,401,67+2*bol+136+1,405);
        outtextxy(67+2*bol+136+1,410,"30h");
        line(67+4*bol+2,401,67+4*bol+2,405);
        outtextxy(53+4*bol+2,410,"100h");
    }
    else if(eksenx_max1==10){
        bol=int(570/2) ;
        line(67,401,67,405);
        outtextxy(67,410,"1h");
        line(67+272,401,67+272,405);
        outtextxy(67+272,410,"3h");
        line(637,401,637,405);
        outtextxy(627,410,"10h");
    }
    settxtjustify(0,2);
    outtextxy(310,451,"FREKANS");
    setviewport(1,469,200,479,1);
    setcolor(63);
    setfillstyle(SOLID_FILL,getcolor());
    bar(0,0,60,10);
    setcolor(0);
    outtextxy(10,1,"ÇIKIŞ");
    setcolor(63);
    outtextxy(80,1,"KURSÖR");
    setlinestyle(0,1,1);
    setviewport(67,10,637,400,1);
    setbkcolor( 0 );
    setcolor(58);
    for(eksenx=1;eksenx<eksenx_max;eksenx+=(eksenx_max/200)){
        if(sayac1!=199){
            UKS_line(eksenx,-ekseny[sayac1],eksenx+(eksenx_max/200),-ekseny[sayac1+1]);
            eksx[sayac1]=u_x(eksenx);
        }
        sayac1++;
    }
    setcolor(63);
    do

```

```

{
switch(getch())
{
case 77:
    setviewport(1,469,200,479,1);
    clearviewport();
    setcolor(63);
    setfillstyle(SOLID_FILL,getcolor());
    bar(70,0,130,10);
    setcolor(0);
    outtextxy(80,1,"KURSÖR");
    setcolor(63);
    outtextxy(10,1,"ÇIKIŞ");
    setviewport(67,10,637,400,1);
    setbkcolor( 0 );
    belirle=1;
    break;

case 75 :
    setviewport(1,469,200,479,1);
    clearviewport();
    setcolor(63);
    setfillstyle(SOLID_FILL,getcolor());
    bar(0,0,60,10);
    setcolor(0);
    outtextxy(10,1,"ÇIKIŞ");
    setcolor(63);
    outtextxy(80,1,"KURSÖR");
    setviewport(67,10,637,400,1);
    setbkcolor( 0 );
    belirle=0;
    break;

case CR:
    if(belirle==1){
        setviewport(1,469,200,479,1);
        clearviewport();
        setcolor(63);
        setfillstyle(SOLID_FILL,getcolor());
        bar(0,0,60,10);
        setcolor(0);
        outtextxy(10,1,"ÇIKIŞ");
        setcolor(63);
        setviewport(67,10,637,400,1);
        setbkcolor( 0 );
        ilkcurposx=0;
        ilkcurposy=v_y(-ekseny[0]);
        for(l=0;l<200;l++){
            eksy[l]=v_y(-ekseny[l]);
        }
        boyut=getmaxx();
        ESCTusu=grcursor(fx,fy,ilkcurposx,ilkcurposy,boyut,1,eksy,eksx,
            u_x(eksenx_max),ENTER_KEY,&soncurposx,&soncurposy);
        setviewport(1,469,200,479,1);
        clearviewport();
        setcolor(63);
        setfillstyle(SOLID_FILL,getcolor());
        bar(0,0,60,10);
        setcolor(0);
        outtextxy(10,1,"ÇIKIŞ");
        setcolor(63);
    }
}

```

```

        outtextxy(80,1,"KURSÖR");
        setviewport(67,10,637,400,1);
        setbkcolor( 0 );
        belirle=0;
        ilkcurposx=soncurposx;
        ilkcurposy=soncurposy;
        setviewport(450,440,639,479,0);
        clearviewport();
        setviewport(67,10,637,400,1);
        bitti6=0;
    }
    else{
        bitti6=1;
    }
    break;
}
}while(!bitti6);
closegraph();
}
void spawnle_devrec(void)
{
    int result;

    result = spawnle(P_WAIT, "devrec.exe", NULL, NULL);
    if (result == -1)
    {
        perror("Error from spawnle");
        exit(1);
    }
}
void spawnle_example(void)
{
    int result;

    clrscr();
    result = spawnle(P_WAIT, "bgs.exe", NULL, NULL);
    if (result == -1)
    {
        perror("Error from spawnle");
        exit(1);
    }
}
void spawnle_example1(void)
{
    int result;

    clrscr();
    result = spawnle(P_WAIT, "hgs.exe", NULL, NULL);
    if (result == -1)
    {
        perror("Error from spawnle");
        exit(1);
    }
}
int tusal(void)
{
    int TUS,lo, hi;
    TUS=bioskey(0);
    lo=TUS & 0X00FF;

```

```

hi=(TUS & 0XFF00) >> 8;
return((lo==0) ? hi+256 : lo);
}
main()
{
    int secim ;
    textbackground(1);
    textcolor(15);
    while(1){
        clrscr();
        _setcursortype(_NOCURS);
        secim=menu();
        switch(secim){
            case F1 :
                clrscr(),
                ags();
                break;
            case F2 :
                clrscr();
                spawnle_example1();/*hgs*/
                break;
            case F3 :
                clrscr();
                spawnle_example();/*bgs*/
                break;

            case F4 :
                if((dosya=fopen("tez.hlp","r+t")) == NULL){
                    textbackground(0);
                    clrscr();
                    exit(0);
                }

                clrscr();
                gotoxy(1,25);
                window(1,24,80,26);
                textbackground(0);
                cprintf(" ESC çıkış");
                cprintf(" HOME İlk Sayfa");
                cprintf(" END Son Sayfa");
                cprintf(" PgUp Sayfa Yukarı");
                cprintf(" PgDn Sayfa Aşağı ");
                window(1,24,80,25);
                textbackground(4);
                cprintf(" ESC çıkış");
                cprintf(" HOME İlk Sayfa");
                cprintf(" END Son Sayfa");
                cprintf(" PgUp Sayfa Yukarı");
                cprintf(" PgDn Sayfa Aşağı ");

                window(1,1,80,23);
                textbackground(1);
                sayfa=1;
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                    printf("%s",METIN);
                }
                BITTI3=0;

```

```

do{
    switch(KAR=tusal()) {
        case HOME :
            if(sayfa==1)
                break;
            else
                clrscr();
                window(1,1,80,23);
                textbackground(1);
                fseek(dosya,0L,SEEK_SET);
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                    printf("%s",METIN);
                }
                sayfa=1;
                break;

        case END :
            if(sayfa==6)
                break;
            else
                sayfa=6;
                fseek(dosya,0L,SEEK_SET);
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                }
                gotoxy(1,wherey());
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                }
                gotoxy(1,wherey());
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                }
                gotoxy(1,wherey());
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                }
                gotoxy(1,wherey());
                clrscr();
                window(1,1,80,23);
                textbackground(1);
                for(satir=0 ; satir<23 ; satir++){
                    fgetpos(dosya,&POZIS);
                    fgets(METIN,200,dosya);
                    printf("%s",METIN);
                }
                break;

        case PAGE_DOWN :
            if(sayfa<6){

```

```

gotoxy(1,wherey());
clrscr();
window(1,1,80,23);
textbackground(1);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
printf("%s",METIN);
}
sayfa++;
}
else
    break;
break;
case PAGE_UP :
if(sayfa==2){
fseek(dosya,0L,SEEK_SET);
clrscr();
window(1,1,80,23);
textbackground(1);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
printf("%s",METIN);
}
sayfa--;
}
else if(sayfa==3 ){
fseek(dosya,0L,SEEK_SET);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
clrscr();
window(1,1,80,23);
textbackground(1);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
printf("%s",METIN);
}
sayfa--;
}
else if(sayfa==4){
fseek(dosya,0L,SEEK_SET);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
clrscr();
window(1,1,80,23);
textbackground(1);

```

```

for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
printf("%s",METIN);
}
sayfa--;
}
else if(sayfa==5){
fseek(dosya,0L,SEEK_SET);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
clrscr();
window(1,1,80,23);
textbackground(1);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
printf("%s",METIN);
}
sayfa--;
}
else if(sayfa==6){
fseek(dosya,0L,SEEK_SET);
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
for(satir=0 ; satir<23 ; satir++){
fgetpos(dosya,&POZIS);
fgets(METIN,200,dosya);
}
gotoxy(1,wherey());
clrscr();
window(1,1,80,23);
textbackground(1);

```

```

for(satir=0 ; satir<23 ; satir++){
    fgetpos(dosya,&POZIS);
    fgets(METIN,200,dosya);
    printf("%s",METIN);
}
sayfa--;
}
break;

case ESC :
    window(1,1,80,25);
    textbackground(1);
    BITTI3=1;
}

} while(!BITTI3);
fclose(dosya);
break;

case F5 :
    _setcursortype(_NORMALCURSOR);
    textbackground(0);
    textcolor(7);
    clrscr();
    exit(0);
default :
    printf("\a");
}
}

menu()
{
    int h;
    window(1,1,80,25);
    clrscr();
    gotoxy(1,7) ;
    printf("┌───────────────────────────────────────────┐\n");
    gotoxy(1,8) ;
    printf("│                                                                 │\n");
    gotoxy(1,9) ;
    printf("│          İ          .[F1].....AGS          .          İ          │\n");
    gotoxy(1,10);
    printf("│                                                                 │\n");
    gotoxy(1,11);
    printf("│          T          .[F2].....YGS          .          │\n");
    gotoxy(1,12);
    printf("│          T          .[F3].....BGS          .          T          │\n");
    gotoxy(1,13);
    printf("│                                                                 │\n");
    gotoxy(1,14);
    printf("│          Ü          .[F4].....YARDIM          .          │\n");
    gotoxy(1,15);
    printf("│          Ü          .[F5].....ÇIKIŞ          .          Ü          │\n");
    gotoxy(1,16);
    printf("│                                                                 │\n");
    gotoxy(1,17);
    printf("└───────────────────────────────────────────┘\n");
    gotoxy(33,20);
    h=tusal();
    return h;
}

void ags()
{
    char kr,boyut[2],a1[30],isim1[30],uz[]=".ags",uz1[]=".adt",ad[1][20];
    do{
        float C[10][8],h[20],T[20][200];

```

```

float Ap=0,As=0,Fc=0,Fp=0,Fs=0,Kv=0,enkucuk,carpan1,carpan2, min=0, f, w1;
long float Fctp,Fcts,Wcutoff,Bk,x,y,u,v,a,b,b1,c,d,m,n,p,r,Bk1, m1,n1,p1,r1; long float
w,l,t,k1, k2,k3, k4,T1,T2,max,max1,k5,k6,k7,k8;
int text=0,i,k,j,N,sayac,sayac1,sayac2,sayac3,sayac4,sayac5,sayac6,sayac7;
int kr1,sayac8, sayac9,sayac10,sayac13;
long int g,s;

```

```

BITTI=0;
BITTI1=0;
BITTI2=0;
clrscr();
_setcursortype(_NORMALCURSOR);
gotoxy(10,5) ; printf("
gotoxy(10,6) ; printf("
gotoxy(10,7) ; printf("
gotoxy(10,8) ;printf("
gotoxy(10,9) ;printf("
gotoxy(10,10) ;printf("
gotoxy(10,11) ;printf("
gotoxy(10,12) ;printf("
gotoxy(10,13) ;printf("
gotoxy(10,14) ;printf("
gotoxy(10,15) ;printf("
window(62,1,79,4);
gotoxy(62,2);cprintf("
gotoxy(62,3);cprintf("
gotoxy(62,4);cprintf("
window(62,4,79,7);
gotoxy(62,5);cprintf("
gotoxy(62,6);cprintf("
gotoxy(62,7);cprintf("
window(62,7,79,10);
gotoxy(62,8) ;cprintf("
gotoxy(62,9) ;cprintf("
gotoxy(62,10);cprintf("
window(62,10,79,13);
gotoxy(62,11);cprintf("
gotoxy(62,12);cprintf("
gotoxy(62,13);cprintf("
window(62,13,79,16);
gotoxy(62,14);cprintf("
gotoxy(62,15);cprintf("
gotoxy(62,16);cprintf("
window(1,1,61,25);
gotoxy(20,8);
sayac8=0;
do{

    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=1;
            goto exit;

        case BACK_SPACE :
            if(wherex()==20)
                break;
            printf("\b");

```

```

sayac9=wherex();
clreol();
gotoxy(45,wherey());printf(" \| ");
gotoxy(sayac9,wherey());
a1[sayac8-1]=0;
sayac8--;
b1=atof(a1);
break;

```

case CR :

```

if(wherex()==20){
    printf("\a");
    break;
}
else{
    if(wherey()==8) {
        b1=atof(a1);
        Ap=b1;
    }
    else if(wherey()==9){
        b1=atof(a1);
        As=b1;
    }
    else if(wherey()==10){
        b1=atof(a1);
        Fp=b1;
    }
    else if(wherey()==11) {
        b1=atof(a1);
        Fs=b1;
    }
    else if(wherey()==12) {
        b1=atof(a1);
        Fc=b1;
    }
    else if(wherey()==13) {
        b1=atof(a1);
        Kv=b1;
    }
    else if(wherey()==14) {
        b1=atof(a1);
        min=b1;
    }
}

for(j=0;j<sayac8+1;j++)
    a1[j]=0;
sayac8=0;

if(wherey()<=14)
    gotoxy(20,wherey()+1);
if(wherey()==15)
    gotoxy(20,8);

```

```

        break;
    }
default :
    if(KAR>=48 && KAR<=57 || KAR==46){

        a1[sayac8]=KAR;
        sayac8++;
        gotoxy(wherex(),wherey());printf("%c",KAR);
        break;

    }
    else{

        printf("\a");
        break;

    }
case YUKARI_TUSU :

    if(wherey()==8)
        break;
    else
        gotoxy(wherex(),wherey()-1);
        break;
case ASAGI_TUSU :

    if(wherey()==14)
        break;
    else
        gotoxy(wherex(),wherey()+1);
        break;

case DEL :

    if(wherey()==8){
        gotoxy(20,8);
        clreol();
        gotoxy(45,8);printf(" || ");
        gotoxy(20,8);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherey()==9){
        gotoxy(20,9);
        clreol();
        gotoxy(45,9);printf(" || ");
        gotoxy(20,9);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherey()==10){
        gotoxy(20,10);
        clreol();
        gotoxy(45,10);printf(" || ");
        gotoxy(20,10);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;
    }

```

```

        sayac8=0;
        break;
    }
    else if(wherex()==11){
        gotoxy(20,11);
        clreol();
        gotoxy(45,11);printf(" || ");
        gotoxy(20,11);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==12){
        gotoxy(20,12);
        clreol();
        gotoxy(45,12);printf(" || ");
        gotoxy(20,12);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==13) {
        gotoxy(20,13);
        clreol();
        gotoxy(45,13);printf(" || ");
        gotoxy(20,13);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==14){
        gotoxy(20,14);
        clreol();
        gotoxy(45,14);printf(" || ");
        gotoxy(20,14);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
}
case F7 :

    for(j=0;j<30;j++)

        isim[j]=0;

    sayac8=0;
    yuk:
    BITTI4=0;
    sayac8=0;
    window(62,16,79,19);
    gotoxy(62,17);cprintf(" ");
    gotoxy(62,18);cprintf(" F6 DOSYA LİSTESİ ");
    gotoxy(62,19);cprintf(" ");
    window(1,1,61,25);
    gotoxy(10,2);printf(" ");
    gotoxy(10,3);printf(" DOSYA İSMİ: ");
    gotoxy(10,4);printf(" ");

```

```

gotoxy(22,3);
do{
switch(KAR=tusal())
{
case F7 :
        window(62,16,79,19);
        clrscr();
        window(1,1,61,25);
        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        gotoxy(20,8);
        BITTI4=1;
        break;

case F6 :
        ara=0;
        directory();
        window(1,1,61,25);
        gotoxy(22,3);
        goto g;

default :
        if(KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
        KAR>=97 && KAR<=122){
                isim[sayac8]=KAR;
                sayac8++;
                printf("%c",KAR);
                break;
        }
        else{
                printf("\a");
                break;
        }
case BACK_SPACE :
        if(wherex()==22)
                break;
        printf("\b");
        sayac9=wherex();
        clreol();
        gotoxy(45,wherey());printf(" || ");
        gotoxy(sayac9,wherey());
        isim[sayac8-1]=0;
        sayac8--;
        break;

case CR :
        g :
        window(62,16,79,19);
        clrscr();
        window(1,1,61,25);
        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        strcpy(isim1,isim);
        strcat(isim,uz);
        if((dosya=fopen(isim,"rb"))==NULL){
                for(j=0;j<sayac8+5;j++)
                        isim[j]=0;
                sayac8=0;
                goto yuk;
        }
}
}

```

```

    }
    fscanf(dosya,"%f\n%f\n%f\n%f\n%f\n%f\n%f",&Ap,&As,&Fp,
                                                &Fs,&Fc,&Kv,&min);

    gotoxy(20,8);clreol();
    gotoxy(45,8);printf(" || ");
    gotoxy(20,8);printf("%f",Ap);
    gotoxy(20,9);clreol();
    gotoxy(45,9);printf(" || ");
    gotoxy(20,9);printf("%f",As);
    gotoxy(20,10);clreol();
    gotoxy(45,10);printf(" || ");
    gotoxy(20,10);printf("%f",Fp);
    gotoxy(20,11);clreol();
    gotoxy(45,11);printf(" || ");
    gotoxy(20,11);printf("%f",Fs);
    gotoxy(20,12);clreol();
    gotoxy(45,12);printf(" || ");
    gotoxy(20,12);printf("%f",Fc);
    gotoxy(20,13);clreol();
    gotoxy(45,13);printf(" || ");
    gotoxy(20,13);printf("%f",Kv );
    gotoxy(20,14);clreol();
    gotoxy(45,14);printf(" || ");
    gotoxy(20,14);printf("%f",min);
    gotoxy(20,8);
    fclose(dosya);

    for(j=0;j<30;j++)
        isim[j]=0;
    BITTI4=1;
    break;
}
}while(!BITTI4);
break;
case F8 :
    for(j=0;j<30;j++)
        isim[j]=0;

    sak :
    BITTI5=0;
    sayac8=0;
    gotoxy(10,2);printf(" ");
    gotoxy(10,3);printf("DOSYA İSMİ:");
    gotoxy(10,4);printf(" ");
    gotoxy(22,3);
    do{
        switch(KAR=tusal())
        {
            case F8 :
                gotoxy(10,2);clreol();
                gotoxy(10,3);clreol();
                gotoxy(10,4);clreol();
                BITTI5=1;
                break;

            default :
                if(KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
                    KAR>=97 && KAR<=122){
                    isim[sayac8]=KAR;
                    sayac8++;
                    printf("%c",KAR);
                }
            }
        }
    }

```

```

        break;
    }
    else{
        printf("\a");
        break;
    }
case BACK_SPACE :
    if(wherex()==22)
        break;
    printf("\b");
    sayac9=wherex();
    clreol();
    gotoxy(45,wherey());printf(" || ");
    gotoxy(sayac9,wherey());
    isim[sayac8-1]=0;
    sayac8--;
    break;
case CR :

    gotoxy(10,2);clreol();
    gotoxy(10,3);clreol();
    gotoxy(10,4);clreol();
    strcpy(isim1,isim);
    strcat(isim,uz);
    gotoxy(20,8);
    if((dosya=fopen(isim,"wb"))==NULL){
        for(j=0;j<sayac8+5;j++)
            isim[j]=0;
        sayac8=0;
        goto sak;
    }
    fprintf(dosya,"%f\n%f\n%f\n%f\n%f\n%f\n%f\n",Ap,As,Fp
        ,Fs,Fc,Kv,min);

    fclose(dosya);
    for(j=0;j<30;j++)
        isim[j]=0;
    BITTI5=1;
    break;
}
}while(!BITTI5);
gotoxy(20,8);
break;
case F10 :
    BITTI1=1;
    goto exit;

case F9 :
    BITTI=1;
    break;
}
}while(!BITTI);

window(1,1,80,25);
textbackground(0);
clrscr();
window(1,1,80,24);
textbackground(1);
clrscr();

```

```

window(1,24,80,25);
textbackground(4);
cprintf(" ESC PAR MENÜ");
cprintf(" F1 DEVRE YAPISI");
cprintf(" F2 FREKANS EĞRİSİ");
cprintf(" F10 ANA MENÜ ");
window(1,1,80,23);
textbackground(1);
clrscr();
_setcursortype(_NOCURSOR);
if(Fp>=Fs || Ap>=1 || As>=Ap){
window(22,10,58,15);
textbackground(4);
gotoxy(22,11);cprintf(" ");
gotoxy(22,12);cprintf(" GİRİLEN KARAKTERİSTİKLER ALÇAK ");
gotoxy(22,13);cprintf(" GEÇİREN FİLTREYİ GERÇEKLEMEZ. ");
gotoxy(22,14);cprintf(" ");
goto menu;
}
if((2*Fs)>=Fc){
window(22,10,58,15);
textbackground(4);
gotoxy(22,11);cprintf(" ");
gotoxy(22,12);cprintf(" ÖRNEKLEME FREKANSINIZ UYGUN ");
gotoxy(22,13);cprintf(" OLMADIĞINDAN ÖRTÜŞME OLUŞTU. ");
gotoxy(22,14);cprintf(" ");
goto menu;
}
x=(PI*Fp)/Fc;
y=(PI*Fs)/Fc;
Fctp=(Fc/PI)*tan(x);
Fcts=(Fc/PI)*tan(y);
v=((1/(As*As))-1)/((1/(Ap*Ap))-1);
a=Fcts/Fctp;
u=log(v)/(2*log(a));
N=ceil(u);
b=((1/(Ap*Ap))-1);
c=0.5/N;
d=(pow(b,c));
Wcutoff=((2*PI*Fctp)/d);
l=Kv/20;
t=pow(10,l);
i= N%2;
if( i==0 ) {
k=N/2;
cprintf(" %d \r\n",k);
cprintf("H(Z)=TT Hj(Z) ; N=%d. DERECE AGS\r\n",N );
cprintf(" j=1 \r\n" );
strcat(isim1,uz1);
if((dosya=fopen(isim1,"wb"))==NULL){
text=1;
}
Bk1=2*sin((PI)/(2*N));
m1=((4*Fc*Fc)+(2*Fc*Bk1*Wcutoff)+(Wcutoff*Wcutoff));
n1=(Wcutoff*Wcutoff)/m1;
p1=((2*Wcutoff*Wcutoff)-(8*Fc*Fc))/m1;
r1=((4*Fc*Fc)-(2*Bk1*Fc*Wcutoff)+(Wcutoff*Wcutoff))/m1;
if(r1<0){
window(22,10,66,15);

```

```

textbackground(4);
gotoxy(22,11);
cprintf(" ");
gotoxy(22,12);
cprintf(" || -2 || ");
gotoxy(22,13);
cprintf(" || z katsayısı negatif. Bu bölgede kalan || ");
gotoxy(22,14);
cprintf(" || fonksiyonlar bu devre ile gerçekleşemezler. || ");
gotoxy(22,15);
cprintf(" ");
goto menu;
}
cprintf("          -1 2          \r\n");
cprintf("          % f(1+Z )          \r\n",t*n1);
cprintf("H1(Z)=-----\r\n");
cprintf("          -1 -2          \r\n");
cprintf("          1 %+fZ %+fZ          \r\n",p1,r1);
if(text==0){
fprintf(dosya,"          %d          \n",k);
fprintf(dosya,"H(Z)=TT Hj(Z) ; N=%d. DERECE AGS\n",N);
fprintf(dosya,"          j=1          \n");
fprintf(dosya,"          -1 2          \n");
fprintf(dosya,"          % f(1+Z )          \n",t*n1);
fprintf(dosya,"H1(Z)=-----\n");
fprintf(dosya,"          -1 -2          \n");
fprintf(dosya,"          1 %+fZ %+fZ          \n",p1,r1);
}
/*NORMLIZASYON KATSAYISININ BULUNDUGU BOLUM*/

f=0;
k1=t*n1*(1+cos(2*2*PI*f/Fc)+2*cos(2*PI*f/Fc)) ;
k2=t*n1*(sin(2*2*PI*f/Fc)+2*sin(2*PI*f/Fc)) ;
k3=(1+(r1*cos(2*2*PI*f/Fc)))+(p1*cos(2*PI*f/Fc)) ;
k4=(r1*sin(2*2*PI*f/Fc))+(p1*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
sayac13=0;
max=T1;
for(f=0;f<Fc/2;f+=(Fc/2)/200){
k1=t*n1*(1+cos(2*2*PI*f/Fc)+(2*cos(2*PI*f/Fc))) ;
k2=t*n1*(sin(2*2*PI*f/Fc)+(2*sin(2*PI*f/Fc))) ;
k3=(1+(r1*cos(2*2*PI*f/Fc)))+(p1*cos(2*PI*f/Fc)) ;
k4=(r1*sin(2*2*PI*f/Fc))+(p1*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
T[0][sayac13]=T1;
fx[sayac13]=f;
fy[sayac13]=T1;
sayac13++;
if(T1>max)
max=T1;
}

sayac13=0;
f=0;
k5=((1-cos(2*PI*f/Fc))*(((t*n1*(1+p1+r1))/r1)-4*t*n1))-4*t*n1*((1/r1)-1);
k6=sin(2*PI*f/Fc)*(t*n1*((1+p1+r1)/r1)-4*t*n1);
k7=(1+(p1*cos(2*PI*f/Fc)))+(r1*cos(2*2*PI*f/Fc))/r1;
k8=((p1*sin(2*PI*f/Fc))+(r1*sin(2*2*PI*f/Fc)))/r;
T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)));

```

```

max1=T2;
for(f=0;f<Fc/2;f+=(Fc/2)/200){

k5=((1-cos(2*PI*f/Fc))*(((t*n1*(1+p1+r1))/r1)-4*t*n1))-4*t*n1*((1/r1)-1);
k6=sin(2*PI*f/Fc)*(t*n1*((1+p1+r1)/r1)-4*t*n1);
k7=(1+(p1*cos(2*PI*f/Fc))+(r1*cos(2*2*PI*f/Fc)))/r1;
k8=((p1*sin(2*PI*f/Fc))+(r1*sin(2*2*PI*f/Fc)))/r1;
T2=(sqrt((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)));
if(T2>max1)
    max1=T2;
}
h[1]=max/max1;
/*      SONU      */
C[0][1]=4*t*n1;
C[0][2]=1+(p1)+r1;
C[0][3]=1/h[1];
C[0][4]=(t*n1)/r1;
C[0][5]=(1/r1)/h[1];
C[0][6]=1;
C[0][7]=(1/r1)-1;
enkucuk=C[0][1];
for (sayac=2;sayac<4;sayac++)
    if(C[0][sayac]<enkucuk)
        enkucuk=C[0][sayac];
carpan1=min/enkucuk;
enkucuk=C[0][4];
for (sayac=5;sayac<8;sayac++)
    if(C[0][sayac]<enkucuk)
        enkucuk=C[0][sayac];
carpan2=min/enkucuk;
C[0][1]=C[0][1]*carpan1;
C[0][2]=C[0][2]*carpan1;
C[0][3]=C[0][3]*carpan1;
C[0][4]=C[0][4]*carpan2;
C[0][5]=C[0][5]*carpan2;
C[0][6]=C[0][6]*carpan2;
C[0][7]=C[0][7]*carpan2;

for(j=2;j<=k;j++){
    Bk=2*sin(((2*j-1)*PI)/(2*N));
    m=((4*Fc*Fc)+(2*Fc*Bk*Wcutoff)+(Wcutoff*Wcutoff));
    n=(Wcutoff*Wcutoff)/m;
    p=((2*Wcutoff*Wcutoff)-(8*Fc*Fc))/m;
    r=((4*Fc*Fc)-(2*Bk*Fc*Wcutoff)+(Wcutoff*Wcutoff))/m;
    if(r<0){
        window(22,10,66,15);
        textbackground(4);
        gotoxy(22,11);
        cprintf(" ");
        gotoxy(22,12);
        cprintf(" || -2 || ");
        gotoxy(22,13);
        cprintf(" || z katsayısı negatif. Bu bölgede kalan || ");
        gotoxy(22,14);
        cprintf(" || fonksiyonlar bu devre ile gerçekleştirilemezler. || ");
        gotoxy(22,15);
        cprintf(" ");
        goto menu;
    }
}

```

```

cprintf("                                -1 2          \r\n");
cprintf("                                % f(1+Z )          \r\n",n);
cprintf("H%d(Z)=-----\r\n",j);
cprintf("                                -1      -2          \r\n ");
cprintf("                                1 %+fZ %+fZ          \r\n",p,r);
if(text==0){
fprintf(dosya,"                                -1 2          \n");
fprintf(dosya,"                                % f(1+Z )          \n",n);
fprintf(dosya,"H%d(Z)=-----\n",j);
fprintf(dosya,"                                -1      -2          \n ");
fprintf(dosya,"                                1 %+fZ %+fZ          \n",p,r);
}

/*NORMLIZASYON KATSAYISININ BULUNDUGU BOLUM*/

f=0;
k1=n*(1+cos(2*2*PI*f/Fc)+2*cos(2*PI*f/Fc)) ;
k2=n*(sin(2*2*PI*f/Fc)+2*sin(2*PI*f/Fc)) ;
k3=(1+(r*cos(2*2*PI*f/Fc)))+(p*cos(2*PI*f/Fc)) ;
k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-2][0];
sayac13=0;
max=T1;
for(f=0;f<Fc/2;f+=(Fc/2)/200){
    k1=n*(1+cos(2*2*PI*f/Fc)+(2*cos(2*PI*f/Fc))) ;
    k2=n*(sin(2*2*PI*f/Fc)+(2*sin(2*PI*f/Fc))) ;
    k3=(1+(r*cos(2*2*PI*f/Fc)))+(p*cos(2*PI*f/Fc)) ;
    k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
    T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-2][sayac13];
    T[j-1][sayac13]=T1;
    fx[sayac13]=f;
    fy[sayac13]=T1;
    sayac13++;
    if(T1>max)
        max=T1;
}
sayac13=0;
f=0;
k5=((1-cos(2*PI*f/Fc))*(((n*(1+p+r))/r)-4*n))-4*n*((1/r)-1);
k6=sin(2*PI*f/Fc)*(n*((1+p+r)/r)-4*n);
k7=(1+(p*cos(2*PI*f/Fc)))+(r*cos(2*2*PI*f/Fc))/r;
k8=((p*sin(2*PI*f/Fc)))+(r*sin(2*2*PI*f/Fc))/r;
T2=sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8))*T[j-2][0];
max1=T2;
for(f=0;f<Fc/2;f+=(Fc/2)/200){
    k5=((1-cos(2*PI*f/Fc))*(((n*(1+p+r))/r)-4*n))-4*n*((1/r)-1);
    k6=sin(2*PI*f/Fc)*(n*((1+p+r)/r)-4*n);
    k7=(1+(p*cos(2*PI*f/Fc)))+(r*cos(2*2*PI*f/Fc))/r;
    k8=((p*sin(2*PI*f/Fc)))+(r*sin(2*2*PI*f/Fc))/r;
    T2=(sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8)))*T[j-2][sayac13];
    sayac13++;
    if(T2>max1)
        max1=T2;
}
h[j]=max/max1;
/*      SONU      */
C[j-1][1]=4*n;
C[j-1][2]=1+(p)+r;
C[j-1][3]=1/h[j];

```

```

C[j-1][4]=(n)/r;
C[j-1][5]=(1/r)/h[j];
C[j-1][6]=1;
C[j-1][7]=(1/r)-1;
enkucuk=C[j-1][1];
for (sayac=2;sayac<4;sayac++)
    if(C[j-1][sayac]<enkucuk)
        enkucuk=C[j-1][sayac];
carpan1=min/enkucuk;
enkucuk=C[j-1][4];
for (sayac=5;sayac<8;sayac++)
    if(C[j-1][sayac]<enkucuk)
        enkucuk=C[j-1][sayac];
carpan2=min/enkucuk;
C[j-1][1]=C[j-1][1]*carpan1;
C[j-1][2]=C[j-1][2]*carpan1;
C[j-1][3]=C[j-1][3]*carpan1;
C[j-1][4]=C[j-1][4]*carpan2;
C[j-1][5]=C[j-1][5]*carpan2;
C[j-1][6]=C[j-1][6]*carpan2;
C[j-1][7]=C[j-1][7]*carpan2;
for(g=0;g<10e5;g++)
    s+=g;
}
cprintf("\r\n");
sayac4=0;
for(sayac3=3;sayac3<1e12; sayac3=sayac3+3){
    for(sayac2=1 ;sayac2<8 ;sayac2++){
        for (sayac1=sayac4; sayac1<sayac3 ; sayac1++){
            if(sayac1<k){
                cprintf("C[%d][%d]=%10.5fpf ", sayac1+1, sayac2,C[sayac1][sayac2]);
                if(text==0){
                    fprintf(dosya,"C[%d][%d]=%10.5fpf ",sayac1,sayac2, C[sayac1][sayac2]);
                }
            }
        }
        cprintf("\r\n");
        if(text==0){
            fprintf(dosya,"\n");
        }
    }
    cprintf("\r\n");
    if(text==0){
        fprintf(dosya,"\n");
    }
    sayac4=sayac4+3;
    if(sayac1>=k) goto menu;
}
fclose(dosya);
for(j=0;j<30;j++)
    isim1[j]=0;
}
else {
    k=(N-1)/2;
    cprintf("                %d                \r\n",k);
    cprintf("H(Z)=Ho(Z)*TT  Hj(Z) ; N=%d. DERECE AGS\r\n",N);
    cprintf("                j=1                \r\n ");
    strcat(isim1,uz1);
    if((dosya=fopen(isim1,"wb"))==NULL){

```

```

text=1;
}
q=Wcutoff/(2*Fc+Wcutoff);
w=(Wcutoff-2*Fc)/(Wcutoff+2*Fc);
cprintf("          -1      \n"),
cprintf("      %f(1+Z )      \n",q*t);
cprintf("Ho(Z) =----- \n");
cprintf("          -1\n");
cprintf("      1%+fZ      \n",w);
if(text==0){
fprintf(dosya,"          %d      \n",k);
fprintf(dosya,"H(Z)=Ho(Z)*TT  Hj(Z) ; N=%d. DERECE AGS\n",N);
fprintf(dosya,"          j=1      \n");

fprintf(dosya,"          -1      \n"),
fprintf(dosya,"      %f(1+Z )      \n",q*t);
fprintf(dosya,"Ho(Z) =----- \n");
fprintf(dosya,"          -1      \n");
fprintf(dosya,"      1%+fZ      \n",w);
}
sayac13=0;
f=0;
k1=q*t*(1+cos(2*PI*f/Fc));
k2=q*t*(sin(2*PI*f/Fc));
k3=1+(w*cos(2*PI*f/Fc));
k4=w*sin(2*PI*f/Fc);
T1=sqrt(((k1*k1)+(k2*k2))/((k3*k3)+(k4*k4)));

max=T1;
for(f=0;f<Fc/2;f+=(Fc/2)/200){
k1=q*t*(1+cos(2*PI*f/Fc));
k2=q*t*(sin(2*PI*f/Fc));
k3=1+(w*cos(2*PI*f/Fc));
k4=w*sin(2*PI*f/Fc);
T1=sqrt(((k1*k1)+(k2*k2))/((k3*k3)+(k4*k4)));
T[0][sayac13]=T1;
fx[sayac13]=f;
fy[sayac13]=T1;
sayac13++;
if(T1>max)
max=T1;
}
C[0][3]=1;
C[0][2]=(1-(w))/(1+(w));
C[0][1]=(C[0][2]+C[0][3])*q*t;
enkucuk=C[0][1];
for (sayac=1;sayac<4;sayac++)
if(C[0][sayac]<enkucuk)
enkucuk=C[0][sayac];
carpan1=min/enkucuk;

C[0][4]=0;
C[0][3]=C[0][3]*carpan1;
C[0][2]=C[0][2]*carpan1;
C[0][1]=C[0][1]*carpan1;
C[0][5]=0 ;
C[0][6]=0 ;
C[0][7]=0 ;

```



```

sayac13=0;
k5=((1-cos(2*PI*f/Fc))*(((n*(1+p+r))/r)-4*n))-4*n*((1/r)-1);
k6=sin(2*PI*f/Fc)*(n*((1+p+r)/r)-4*n);
k7=(1+(p*cos(2*PI*f/Fc)+(r*cos(2*PI*f/Fc)))/r;
k8=((p*sin(2*PI*f/Fc)+(r*sin(2*PI*f/Fc)))/r;
T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)))*T[j-1][0];
max1=T2;
for(f=0;f<Fc/2;f+=(Fc/2)/200){
    k5=((1-cos(2*PI*f/Fc))*(((n*(1+p+r))/r)-4*n))-4*n*((1/r)-1);
    k6=sin(2*PI*f/Fc)*(n*((1+p+r)/r)-4*n);
    k7=(1+(p*cos(2*PI*f/Fc)+(r*cos(2*PI*f/Fc)))/r;
    k8=((p*sin(2*PI*f/Fc)+(r*sin(2*PI*f/Fc)))/r;
    T2=(sqrt(((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)))*T[j-1][sayac13];
    sayac13++;
    if(T2>max1)
        max1=T2;
}
h[j]=max/max1;
/*      SONU      */
C[j][1]=4*n;
C[j][2]=1+(p)+r;
C[j][3]=1/h[j];
C[j][4]=(n)/r;
C[j][5]=(1/r)/h[j];
C[j][6]=1;
C[j][7]=(1/r)-1;
enkucuk=C[j][1];
for (sayac=2;sayac<4;sayac++)
    if(C[j][sayac]<enkucuk)
        enkucuk=C[j][sayac];
carpan1=min/enkucuk;
enkucuk=C[j][4];
for (sayac=5;sayac<8;sayac++)
    if(C[j][sayac]<enkucuk)
        enkucuk=C[j][sayac];
carpan2=min/enkucuk;
C[j][1]=C[j][1]*carpan1;
C[j][2]=C[j][2]*carpan1;
C[j][3]=C[j][3]*carpan1;
C[j][4]=C[j][4]*carpan2;
C[j][5]=C[j][5]*carpan2;
C[j][6]=C[j][6]*carpan2;
C[j][7]=C[j][7]*carpan2;
for(g=0;g<10e5;g++)
    s+=g;
}
cprintf("\r\n");
sayac4=0;
for(sayac3=3;sayac3<1e12; sayac3=sayac3+3){
    for(sayac2=1 ;sayac2<8 ;sayac2++){
        for (sayac1=sayac4; sayac1<sayac3 ; sayac1++){
            if(sayac1<k+1){
                cprintf("C[%d][%d]=%10.5fpf ",sayac1,sayac2,C[sayac1][sayac2]);
                if(text==0){
                    fprintf(dosya,"C[%d][%d]=%10.5fpf ",sayac1,sayac2,C[sayac1][sayac2]);
                }
            }
        }
    }
}

```

```

        cprintf("\r\n");
        if(text==0){
            fprintf(dosya,"\n");
        }
    }
    cprintf("\r\n");
    if(text==0){
        fprintf(dosya,"\n");
    }
    sayac4=sayac4+3;
    if(sayac1>=k+1) goto menu;
}
fclose(dosya);
for(j=0;j<30;j++)
    isim1[j]=0;
}
menu:
do{
    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=0;
            BITTI2=1;
            window(1,1,80,25);
            textbackground(1);
            clrscr();
            break;

        case F10 :
            BITTI1=1;
            BITTI2=1;
            window(1,1,80,25);
            textbackground(1);
            clrscr();
            break;

        case F1 :
            gettext(1,1,80,25,sakla);
            spawnle_devrec();
            _setcursortype(_NOCURSOR);
            puttext(1,1,80,25,sakla);
            break;

        case F2 :
            sayac13=0;
            eksenx_max=Fc/2;
            if((Fc/2)<=10 && (Fc/2)>0)
                eksenx_max1=10;
            else if((Fc/2)<=100 && (Fc/2)>10)
                eksenx_max1=100;
            else if((Fc/2)<=1000 && (Fc/2)>100)
                eksenx_max1=1000;
            else if((Fc/2)<=10000 && (Fc/2)>1000)
                eksenx_max1=10000;
            else if((Fc/2)<=100000 && (Fc/2)>10000)
                eksenx_max1=100000;
            else if((Fc/2)<=1000000 && (Fc/2)>100000)
                eksenx_max1=1000000;
            else
                eksenx_max1=Fc/2;
            ekseny_max=max;
            for(sayac13=0;sayac13<200;sayac13++){

```

```

        if(N%2==0)
            ekseny[sayac13]=T[k-1][sayac13];
        else
            ekseny[sayac13]=T[k][sayac13];
        }
        gettext(1,1,80,25,sakla);
        grafik();
        _setcursortype(_NOCURS);
        puttext(1,1,80,25,sakla);
        break;
    }
} while(!BITTI2);
exit :
} while(!BITTI1);
}
void directory()
{
    struct find_t ffbk;
    int i,j,done,bitti,filecount;
    char ad[2000][30],b[]=".";
    for(j=0;j<30;j++)
        isim[j]=0;
    window(1,1,80,25);
    textbackground(1);
    if(ara==0)
        done = _dos_findfirst("*.ags",_A_NORMAL,&ffb);
    else if(ara==1)
        done = _dos_findfirst("*.ygs",_A_NORMAL,&ffb);
    i=0;
    while(!done){
        strcpy(ad[i],ffb.name);
        i++;
        done = _dos_findnext(&ffb);
    }
    filecount=i;
    _setcursortype(_NOCURS);
    gotoxy(10,16);printf(" ");
    gotoxy(10,17);printf(" ");
    gotoxy(10,18);printf(" ");
    gotoxy(10,19);printf(" ");
    gotoxy(10,20);printf(" ");
    gotoxy(10,21);printf(" ");
    gotoxy(10,22);printf(" ");
    gotoxy(10,23);printf(" ");
    gotoxy(10,24);printf(" ");
    gotoxy(10,25);printf(" ");
    window(11,17,29,24);
    bitti=0;
    gotoxy(1,1);
    if(filecount==0)
        goto git;
    else if(filecount<7){
        for(j=0;j<filecount;j++)
            cprintf(" %-18s",ad[j]);
    }
    else{
        for(j=0;j<7;j++)
            cprintf(" %-18s",ad[j]);
    }
}

```

```

textattr(15+((16)<<2));
gotoxy(1,1);
cprintf(" %-17s",ad[0]);
i=0;
do{
    switch(KAR=tusal()){

    case ASAGI_TUSU :

        if(i!=filecount-1){

            i++;
            gotoxy(1,wherey());
            textattr(23+((24)<<5));
            textcolor(15);
            cprintf(" %-18s",ad[i-1]);
            gotoxy(1,wherey());
            textattr(15+((16)<<2));
            cprintf(" %-17s",ad[i]);
            break;

        }
        else {
            break;
        }

    case YUKARI_TUSU :
        if(i==0)break;
        if(wherey()==1){
            gotoxy(1,1);
            textattr(15+((16)<<2));
            cprintf(" %-17s",ad[i-1]);
            textattr(23+((24)<<5));
            textcolor(15);
            gotoxy(1,2);cprintf(" %-17s",ad[i]);
            gotoxy(1,3);cprintf(" %-17s",ad[i+1]);
            gotoxy(1,4);cprintf(" %-17s",ad[i+2]);
            gotoxy(1,5);cprintf(" %-17s",ad[i+3]);
            gotoxy(1,6);cprintf(" %-17s",ad[i+4]);
            gotoxy(1,7);cprintf(" %-17s",ad[i+5]);
            gotoxy(1,8);cprintf(" %-17s",ad[i+6]);
            gotoxy(1,1);
            i--;
            break;

        }
        else{
            gotoxy(1,wherey());
            textattr(23+((24)<<5));
            textcolor(15);
            cprintf(" %-17s",ad[i]);
            gotoxy(1,wherey()-1);
            textattr(15+((16)<<2));
            cprintf(" %-17s",ad[i-1]);
            i--;
            break;
        }

    case CR :

        strcpy(isim,ad[i]);
        strtok(isim,b);
        bitti=1;
        break;

```

```

    case ESC :
        don :
            bitti=1;
            break;
        default :
            bitti=0;
            break;
    }
} while(!bitti);
git :
window(10,16,60,25);
textbackground(1);
textattr(23+((24)<< 5));
textcolor(15);
clrscr();
_setcursortype(_NORMALCURSOR);
}

```

HGS.CPP

```

#include<stdio.h>
#include<graphics.h>
#include<process.h>
#include<complex.h>
#include<math.h>
#include<stdlib.h>
#include<dos.h>
#include<bios.h>
#include<conio.h>
#include<fcntl.h>
#include<string.h>
#include<sys\stat.h>
#include "cursor.h"
#define PI (float) (3.141592654)
#define CR 13
#define ESC 27
#define BACK_SPACE 8
#define HOME 327
#define END 335
#define YUKARI_TUSU 328
#define ASAGI_TUSU 336
#define SOL_TUSU 331
#define SAG_TUSU 333
#define PAGE_UP 329
#define PAGE_DOWN 337
#define F1 315 /*AGS*/
#define F2 316 /*HGS*/
#define F3 317 /*BGS*/
#define F4 318 /*YARDIM */
#define F5 319 /*ÇIKIŞ*/
#define F6 320
#define F7 321 /*YÜKLEME*/
#define F8 322 /*SAKLAMA*/
#define F9 323 /*HESAPLAMA*/
#define F10 324 /*ANA MENÜYE DÖNÜŞ*/
#define F11 325
#define F12 326
#define DEL 339 /*SATIR SİLME */
FILE *dosya;

```

```

void directory();
void grafik();
int KAR,sayfa,sayfa1,ara;
float eksenx_max,eksenx_max1 ,eksenx,ekseny[200],ekseny_max,fx[210],fy[210];
char BITTI,BITTI1,BITTI2,BITTI3,BITTI4,BITTI5;
char METIN[200],isim[30],sakla[20000];
int satir;
fpos_t POZIS;
int u_x(float u)
{
    return((int)(log10(u)*570*(1/log10(eksenx_max1))));
}
int v_y(float v)
{
    return((int)(390+(v*325)*(120/((ceil(ekseny_max+0.2))*100))));
}
void UKS_line (float u1, float v1, float u2,float v2)
{
    line ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_outtextxy(float u1,float v1)
{
    outtextxy(u_x(u1),v_y(v1),".");
}
void UKS_circle (float u, float v, float r)
{
    circle ( u_x(u), v_y(v), u_x(r) );
}

void grafik()
{
    int GrAdapter,GrMod;
    int hata_kodu,bol,sayac1=0,sayac2;
    float ymax,boly;
    int ilkcurposx,ilkcurposy;
    int soncurposx,soncurposy;
    int boyut,bitti6=0,l,belirle=0;
    int ESCTusu,eksy[200],eksx[200];
    char katar[31];
    detectgraph ( &GrAdapter,&GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter , &GrMod ,"" );
    hata_kodu = graphresult();

    if(hata_kodu != grOk)
    {
        printf("grafik hatası : (%d kodlu hata)! %s\n",hata_kodu, grapherrormsg (hata_kodu));
        getch();
        exit(1);
    }
    setlinestyle(3,10,1);
    line(60,9,637,9);
    line(67,9,67,401);
    line(60,401,637,401);
    line(637,9,637,401);
    line(60,87,67,87);
    line(60,166,67,166);
    line(60,244,67,244);

```

```

line(60,323,67,323);
line(60,401,67,401);
ymax=ceil(ekseny_max+0.2);
boly=(float)ymax/5;
sprintf(katar, "%.1f",ymax);
outtextxy(30,7,katar);
sprintf(katar, "%.1f",ymax-boly);
outtextxy(30,85,katar);
sprintf(katar, "%.1f",ymax-(2*boly));
outtextxy(30,164,katar);
sprintf(katar, "%.1f",ymax-(3*boly));
outtextxy(30,242,katar);
sprintf(katar, "%.1f",ymax-(4*boly));
outtextxy(30,321,katar);
outtextxy(40,399,"0");
setlinestyle(0,1,1);
settextjustify(1,2);
if(eksenx_maxl==1000000){
    bol=570/6 ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,162,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol,401,257,405);
    outtextxy(67+2*bol,410,"100h");
    line(67+3*bol,401,352,405);
    outtextxy(67+3*bol,410,"1Kh");
    line(67+4*bol,401,447,405);
    outtextxy(67+4*bol,410,"10Kh");
    line(67+5*bol,401,542,405);
    outtextxy(67+5*bol,410,"100Kh");
    line(67+6*bol,401,637,405);
    outtextxy(49+6*bol,410,"1MGh");
}
if(eksenx_maxl==100000){
    bol=570/5 ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,181,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol,401,295,405);
    outtextxy(67+2*bol,410,"100h");
    line(67+3*bol,401,409,405);
    outtextxy(67+3*bol,410,"1Kh");
    line(67+4*bol,401,523,405);
    outtextxy(67+4*bol,410,"10Kh");
    line(67+5*bol,401,637,405);
    outtextxy(49+5*bol,410,"100Kh");
}
else if(eksenx_maxl==10000){
    bol=int(570/4) ;
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,67+bol,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol+1,401,67+2*bol+1,405);
    outtextxy(67+2*bol+1,410,"100h");
    line(67+3*bol+1,401,67+3*bol+1,405);

```

```

        outtextxy(67+3*bol+1,410,"1Kh");
        line(67+4*bol+2,401,67+4*bol+2,405);
        outtextxy(53+4*bol+2,410,"10Kh");
    }
else if(eksenx_max1==1000){
    bol=int(570/6);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+91,401,67+91,405);
    outtextxy(67+91,410,"3h");
    line(67+2*bol,401,67+2*bol,405);
    outtextxy(67+2*bol,410,"10h");
    line(67+2*bol+91,401,67+2*bol+91,405);
    outtextxy(67+2*bol+91,410,"30h");
    line(67+4*bol,401,67+4*bol,405);
    outtextxy(67+4*bol,410,"100h");
    line(67+4*bol+91,401,67+4*bol+91,405);
    outtextxy(67+4*bol+91,410,"300h");
    line(67+6*bol,401,67+6*bol,405);
    outtextxy(57+6*bol,410,"1Kh");
}
else if(eksenx_max1==100){
    bol=int(570/4);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+136,401,67+136,405);
    outtextxy(67+136,410,"3h");
    line(67+2*bol+1,401,67+2*bol+1,405);
    outtextxy(67+2*bol+1,410,"10h");
    line(67+2*bol+136+1,401,67+2*bol+136+1,405);
    outtextxy(67+2*bol+136+1,410,"30h");
    line(67+4*bol+2,401,67+4*bol+2,405);
    outtextxy(53+4*bol+2,410,"100h");
}
else if(eksenx_max1==10){
    bol=int(570/2);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+272,401,67+272,405);
    outtextxy(67+272,410,"3h");
    line(637,401,637,405);
    outtextxy(627,410,"10h");
}
settextjustify(0,2);
outtextxy(310,451,"FREKANS");
setviewport(1,469,200,479,1);
setcolor(63);
setfillstyle(SOLID_FILL,getcolor());
bar(0,0,60,10);
setcolor(0);
outtextxy(10,1,"ÇIKIŞ");
setcolor(63);
outtextxy(80,1,"KURSÖR");
setlinestyle(0,1,1);
setviewport(67,10,637,400,1);
setbkcolor(0);
setcolor(58);
for(eksenx=1;eksenx<eksenx_max;eksenx+=(eksenx_max/200)){
    if(sayac!=199){

```

```

        UKS_line(eksenx,-ekseny[sayac1],eksenx+(eksenx_max/200),-ekseny[sayac1+1]);
        eksx[sayac1]=u_x(eksenx);
    }
    sayac1++;
}
setcolor(63);
do
{
    switch(getch())
    {
        case 77:
            setviewport(1,469,200,479,1);
            clearviewport();
            setcolor(63);
            setfillstyle(SOLID_FILL,getcolor());
            bar(70,0,130,10);
            setcolor(0);
            outtextxy(80,1,"KURSÖR");
            setcolor(63);
            outtextxy(10,1,"ÇIKIŞ");
            setviewport(67,10,637,400,1);
            setbkcolor( 0 );
            belirle=1;
            break;

        case 75 :
            setviewport(1,469,200,479,1);
            clearviewport();
            setcolor(63);
            setfillstyle(SOLID_FILL,getcolor());
            bar(0,0,60,10);
            setcolor(0);
            outtextxy(10,1,"ÇIKIŞ");
            setcolor(63);
            outtextxy(80,1,"KURSÖR");
            setviewport(67,10,637,400,1);
            setbkcolor( 0 );
            belirle=0;
            break;

        case CR:
            if(belirle==1){
                setviewport(1,469,200,479,1);
                clearviewport();
                setcolor(63);
                setfillstyle(SOLID_FILL,getcolor());
                bar(0,0,60,10);
                setcolor(0);
                outtextxy(10,1,"ÇIKIŞ");
                setcolor(63);
                setviewport(67,10,637,400,1);
                setbkcolor( 0 );
                ilkcurposx=0;
                ilkcurposy=v_y(-ekseny[0]);
                for(l=0;l<200;l++){
                    eksy[l]=v_y(-ekseny[l]);
                }
                boyut=getmaxx();
                ESCTusu=grcursor(fx, fy,ilkcurposx,ilkcurposy,boyut,1,eksy,eksx,
                                u_x(eksenx_max),ENTER_KEY,&soncurposx,&soncurposy);
                setviewport(1,469,200,479,1);
            }
    }
}

```

```

        clearviewport();
        setcolor(63);
        setfillstyle(SOLID_FILL, getcolor());
        bar(0,0,60,10);
        setcolor(0);
        outtextxy(10,1,"ÇIKIŞ");
        setcolor(63);
        outtextxy(80,1,"KURSÖR");
        setviewport(67,10,637,400,1);
        setbkcolor(0);
        belirle=0;
        ilkcurposx=soncurposx;
        ilkcurposy=soncurposy;
        setviewport(450,440,639,479,0);
        clearviewport();
        setviewport(67,10,637,400,1);
        bitti6=0;
    }
    else{
        bitti6=1;
    }
    break;
}
}while(!bitti6);
closegraph();
}
void spawnle_devreh(void)
{
    int result;
    result = spawnle(P_WAIT, "devreh.exe", NULL, NULL);
    if (result == -1)
    {
        perror("Error from spawnle");
        exit(1);
    }
}
int tusal(void)
{
    int TUS, lo, hi;
    TUS=bioskey(0);
    lo=TUS & 0X00FF;
    hi=(TUS & 0XFF00) >> 8;
    return((lo==0) ? hi+256 : lo);
}
void main()
{
    char kr, boyut[2], a1[30], isim1[30], uz[]=".ygs", uz1[]=".ydt";
    do{
        float C[10][7], D[10][7], T[20][200], h[20];
        float Ap=0, As=0, Fp=0, Fs=0, Fc=0, Fp1, Fs1, Kv=0, enkucuk, carpan1, carpan2, min=0, f;
        long float Fctp, Fcts, Wcutoff, Bk, x, y, u, v, a, b, b1, c, d, m, n, p, r, q, w, t, l, m1, n1, p1; long float
        r1, Bk1, k1, k2, k3, k4, T1, T2, max, max1, k5, k6, k7, k8;
        int text=0, i, k, j, N, sayac, sayac1, sayac2, sayac3, sayac4, sayac5, sayac6, sayac7;
        int kr1, sayac8, sayac9, sayac13;
        long int g, s;
        BITTI=0;
        BITTI1=0;
        BITTI2=0;
        clrscr();

```

```

_setcursortype(_NORMALCURSOR);
gotoxy(10,5) ; printf("
gotoxy(10,6) ; printf("
gotoxy(10,7) ; printf("
gotoxy(10,8) ; printf("
gotoxy(10,9) ; printf("
gotoxy(10,10) ; printf("
gotoxy(10,11) ; printf("
gotoxy(10,12) ; printf("
gotoxy(10,13) ; printf("
gotoxy(10,14) ; printf("
gotoxy(10,15) ; printf("
window(62,1,79,4);
gotoxy(62,2);cprintf("
gotoxy(62,3);cprintf("
gotoxy(62,4);cprintf("
window(62,4,79,7);
gotoxy(62,5);cprintf("
gotoxy(62,6);cprintf("
gotoxy(62,7);cprintf("
window(62,7,79,10);
gotoxy(62,8) ;cprintf("
gotoxy(62,9) ;cprintf("
gotoxy(62,10);cprintf("
window(62,10,79,13);
gotoxy(62,11);cprintf("
gotoxy(62,12);cprintf("
gotoxy(62,13);cprintf("
window(62,13,79,16);
gotoxy(62,14);cprintf("
gotoxy(62,15);cprintf("
gotoxy(62,16);cprintf("
window(1,1,61,25);
gotoxy(20,8);
sayac8=0;
do{
    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=1;
            goto exit;
        case BACK_SPACE :
            if(wherex()==20)
                break;

            printf("\b");
            sayac9=wherex();
            clreol();
            gotoxy(45,wherey());printf(" || ");
            gotoxy(sayac9,wherey());
            a1[sayac8-1]=0;
            sayac8--;
            b1=atof(a1);
            break;

        case CR :
            if(wherex()==20){
                printf("\a");
                break;
            }
    }
}

```

```

else{
if(wherex()==8) {
    b1=atoi(a1);
    Ap=b1;
}
else if(wherex()==9){
    b1=atoi(a1);
    As=b1;
}
else if(wherex()==10){
    b1=atoi(a1);
    Fp=b1;
}
else if(wherex()==11) {
    b1=atoi(a1);
    Fs=b1;
}
else if(wherex()==12) {
    b1=atoi(a1);
    Fc=b1;
}
else if(wherex()==13) {
    b1=atoi(a1);
    Kv=b1;
}
else if(wherex()==14) {
    b1=atoi(a1);
    min=b1;
}
for(j=0;j<sayac8+1;j++)
    a1[j]=0;
sayac8=0;
if(wherex()<=14)
    gotoxy(20,wherex()+1);
if(wherex()==15)
    gotoxy(20,8);
break;
}
default :
    if(KAR>=48 && KAR<=57 || KAR==46){

        a1[sayac8]=KAR;
        sayac8++;
        gotoxy(wherex(),wherex());printf("%c",KAR);
        break;

    }
    else{

        printf("\a");
        break;

    }
case YUKARI_TUSU :

    if(wherex()==8)
        break;
    else
        gotoxy(wherex(),wherex()-1);
        break;
case ASAGI_TUSU :

```

```

if(wherex()==14)
    break;
else
    gotoxy(wherex(),wherex()+1);
    break;

case DEL :

    if(wherex()==8){
        gotoxy(20,8);
        clreol();
        gotoxy(45,8);printf(" || ");
        gotoxy(20,8);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==9){
        gotoxy(20,9);
        clreol();
        gotoxy(45,9);printf(" || ");
        gotoxy(20,9);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==10){
        gotoxy(20,10);
        clreol();
        gotoxy(45,10);printf(" || ");
        gotoxy(20,10);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==11){
        gotoxy(20,11);
        clreol();
        gotoxy(45,11);printf(" || ");
        gotoxy(20,11);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==12){
        gotoxy(20,12);
        clreol();
        gotoxy(45,12);printf(" || ");
        gotoxy(20,12);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherex()==13) {

```

```

        gotoxy(20,13);
        clreol();
        gotoxy(45,13);printf(" || ");
        gotoxy(20,13);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;
        sayac8=0;
        break;
    }
    else if(wherex()==14){
        gotoxy(20,14);
        clreol();
        gotoxy(45,14);printf(" || ");
        gotoxy(20,14);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;
        sayac8=0;
        break;
    }
case F7 :

    for(j=0;j<30;j++)
        isim[j]=0;

    sayac8=0;
    yuk1:
    BITTI4=0;
    sayac8=0;
    window(62,16,79,19);
    gotoxy(62,17);cprintf(" ");
    gotoxy(62,18);cprintf(" F6 DOSYA LİSTESİ ");
    gotoxy(62,19);cprintf(" ");
    window(1,1,61,25);
    gotoxy(10,2);printf(" ");
    gotoxy(10,3);printf(" DOSYA İSMİ: ");
    gotoxy(10,4);printf(" ");
    gotoxy(22,3);
    do{
        switch(KAR=tusal())
        {
            case F7 :
                window(62,16,79,19);
                clrscr();
                window(1,1,61,25);
                gotoxy(10,2);clreol();
                gotoxy(10,3);clreol();
                gotoxy(10,4);clreol();
                gotoxy(20,8);
                BITTI4=1;
                break;

            case F6 :
                ara=1;
                directory();
                window(1,1,61,25);
                gotoxy(22,3);
                goto g;

            default :
                if(KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
                    KAR>=97 && KAR<=122){
                    isim[sayac8]=KAR;

```

```

        sayac8++;
        printf("%c",KAR);
        break;
    }
    else{
        printf("\a");
        break;
    }
case BACK_SPACE :
    if(wherex()==22)
        break;
    printf("\b");
    sayac9=wherex();
    clreol();
    gotoxy(45,wherey());printf(" || ");
    gotoxy(sayac9,wherey());
    isim[sayac8-1]=0;
    sayac8--;
    break;
case CR :
    g :
    window(62,16,79,19);
    clrscr();
    window(1,1,61,25);
    gotoxy(10,2);clreol();
    gotoxy(10,3);clreol();
    gotoxy(10,4);clreol();
    strcpy(isim1,isim);
    strcat(isim,uz);
    if((dosya=fopen(isim,"rb"))==NULL){
        for(j=0;j<sayac8+5;j++)
            isim[j]=0;
        sayac8=0;
        goto yuk1;
    }
    fscanf(dosya,"%f\n%f\n%f\n%f\n%f\n%f\n%f\n%f",&Ap,&As,
        &Fp,&Fs,&Fc,&Kv,&min);

    gotoxy(20,8);clreol();
    gotoxy(45,8);printf(" || ");
    gotoxy(20,8);printf("%f",Ap);
    gotoxy(20,9);clreol();
    gotoxy(45,9);printf(" || ");
    gotoxy(20,9);printf("%f",As);
    gotoxy(20,10);clreol();
    gotoxy(45,10);printf(" || ");
    gotoxy(20,10);printf("%f",Fp);
    gotoxy(20,11);clreol();
    gotoxy(45,11);printf(" || ");
    gotoxy(20,11);printf("%f",Fs);
    gotoxy(20,12);clreol();
    gotoxy(45,12);printf(" || ");
    gotoxy(20,12);printf("%f",Fc);
    gotoxy(20,13);clreol();
    gotoxy(45,13);printf(" || ");
    gotoxy(20,13);printf("%f",Kv );
    gotoxy(20,14);clreol();
    gotoxy(45,14);printf(" || ");
    gotoxy(20,14);printf("%f",min);

```

```

gotoxy(20,8);
fclose(dosya);
for(j=0;j<sayac8+5;j++)
    isim[j]=0;

BITTI4=1;
break;
}
}while(!BITTI4);
break;
case F8 :
    for(j=0;j<30;j++)
        isim[j]=0;

    BITTI5=0;
    sayac8=0;
    sak1 :
    gotoxy(10,2);printf("
    gotoxy(10,3);printf("DOSYA İSMİ:");
    gotoxy(10,4);printf("
    gotoxy(22,3);
    do{
    switch(KAR=tusal())
    {
    case F8 :
        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        BITTI5=1;
        break;

    default :
        if(KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
        KAR>=97 && KAR<=122){
            isim[sayac8]=KAR;
            sayac8++;
            printf("%c",KAR);
            break;
        }
        else{
            printf("\a");
            break;
        }
    }
    case BACK_SPACE :
        if(wherex()==22)
            break;
        printf("\b");
        sayac9=wherex();
        clreol();
        gotoxy(45,wherey());printf(" || ");
        gotoxy(sayac9,wherey());
        isim[sayac8-1]=0;
        sayac8--;
        break;

    case CR :

        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        strcpy(isim1,isim);
        strcat(isim,uz);
        gotoxy(20,8);

```

```

        if((dosya=fopen(isim,"wb"))==NULL){
            for(j=0;j<sayac8+5;j++){
                isim[j]=0;
                sayac8=0;
                goto sak1;
            }
        }
        fprintf(dosya,"%f\n%f\n%f\n%f\n%f\n%f\n%f\n",Ap,As,
                                                    Fp,Fs,Fc,Kv,min);
        fclose(dosya);
        for(j=0;j<sayac8+5;j++){
            isim[j]=0;
            BITTI5=1;
            break;
        }
    } while(!BITTI5);
    gotoxy(20,8);
    break;
case F10 :
    BITTI1=1;
    goto exit;

case F9 :
    BITTI=1;
    break;
}
}
while(!BITTI);
window(1,1,80,25);
textbackground(0);
clrscr();
window(1,1,80,24);
textbackground(1);
clrscr();
window(1,24,80,25);
textbackground(4);
cprintf(" ESC PAR MENÜ");
cprintf("  F1 DEVRE YAPISI");
cprintf("  F2 FREKANS EĞRİSİ");
cprintf("  F10 ANA MENÜ ");
window(1,1,80,23);
textbackground(1);
clrscr();
_setcursortype(_NOCURSOR);
if(Fs>=Fp || Ap>=1 || As>=Ap){
    window(22,10,59,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("
    goto menu;
}
if((2*Fp)>=Fc){
    window(22,10,58,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("

```

GİRİLEN KARAKTERİSTİKLER YÜKSEK
GEÇİREN FİLTREYİ GERÇEKLEMEZ.

ÖRNEKLEME FREKANSINIZ UYGUN
OLMADIĞINDAN ÖRTÜŞME OLUŞTU


```

fprintf(dosya,"          -1 2          \n");
fprintf(dosya,"          % f(1-Z )          \n",t*n1);
fprintf(dosya,"H1(Z) = ----- \n");
fprintf(dosya,"          -1      -2          \n ");
fprintf(dosya,"          1 % +fZ % +fZ          \n",p1,r1);
}

```

/*NORMLIZASYON KATSAYISININ BULUNDUGU BOLUM*/

```

f=0;
k1=t*n1*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc));
k2=t*n1*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc));
k3=(1+(r1*cos(2*2*PI*f/Fc))+(p1*cos(2*PI*f/Fc)));
k4=(r1*sin(2*2*PI*f/Fc))+(p1*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
T[0][0]=T1;
fx[sayac13]=f;
fy[sayac13]=T1;
sayac13=1;
max=T1;
for(f=((Fc/2)/200);f<=Fc/2;f+=(Fc/2)/200){
    k1=t*n1*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc));
    k2=t*n1*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc));
    k3=(1+(r1*cos(2*2*PI*f/Fc))+(p1*cos(2*PI*f/Fc)));
    k4=(r1*sin(2*2*PI*f/Fc))+(p1*sin(2*PI*f/Fc));
    T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
    T[0][sayac13]=T1;
    fx[sayac13]=f;
    fy[sayac13]=T1;
    sayac13++;
    if(T1>max)
        max=T1;
}
f=0;
k5=t*n1*(1+p1+r1)*(1-cos(2*PI*f/Fc));
k6=t*n1*(1+p1+r1)*sin(2*PI*f/Fc);
k7=(1+(p1*cos(2*PI*f/Fc))+(r1*cos(2*2*PI*f/Fc)));
k8=((p1*sin(2*PI*f/Fc))+(r1*sin(2*2*PI*f/Fc)));
T2=sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8));
max1=T2;
for(f=1+(Fc/400);f<=Fc/2;f+=Fc/400){
    k5=t*n1*(1+p1+r1)*(1-cos(2*PI*f/Fc));
    k6=t*n1*(1+p1+r1)*sin(2*PI*f/Fc);
    k7=(1+(p1*cos(2*PI*f/Fc))+(r1*cos(2*2*PI*f/Fc)));
    k8=((p1*sin(2*PI*f/Fc))+(r1*sin(2*2*PI*f/Fc)));
    T2=(sqrt((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)));
    if(T2>max1)
        max1=T2;
}
h[1]=max/max1;
/*      SONU      */
C[0][0]=1+p1+r1;
C[0][1]=1/h[1];
C[0][2]=(1/r1)/h[1];
C[0][3]=1;
C[0][4]=(1/r1)-1;
C[0][5]=(t*n1)/r1;

enkucuk=C[0][0];

```

```

for (sayac=1;sayac<2;sayac++)
    if(C[0][sayac]<enkucuk)
        enkucuk=C[0][sayac];
carpan1=min/enkucuk;
enkucuk=C[0][2];
for (sayac=3;sayac<6;sayac++)
    if(C[0][sayac]<enkucuk)
        enkucuk=C[0][sayac];
carpan2=min/enkucuk;
D[0][1]=C[0][5]*carpan2;
D[0][2]=C[0][0]*carpan1;
D[0][3]=C[0][1]*carpan1;
D[0][4]=C[0][2]*carpan2;
D[0][5]=C[0][3]*carpan2;
D[0][6]=C[0][4]*carpan2;
C[0][1]=D[0][1];
C[0][2]=D[0][2];
C[0][3]=D[0][3];
C[0][4]=D[0][4];
C[0][5]=D[0][5];
C[0][6]=D[0][6];
for(j=2;j<=k;j++){
    Bk=2*sin(((2*j-1)*PI)/(2*N));
    m=((4*Fc*Fc)+(2*Fc*Bk*Wcutoff)+(Wcutoff*Wcutoff));
    n=(Wcutoff*Wcutoff)/m;
    p=((8*Fc*Fc)-(2*Wcutoff*Wcutoff))/m;
    r=((4*Fc*Fc)-(2*Bk*Fc*Wcutoff)+(Wcutoff*Wcutoff))/m;
    if(r<0){
        window(22,10,66,15);
        textbackground(4);
        gotoxy(22,11);
        cprintf(" ");
        gotoxy(22,12);
        cprintf("  -2 ");
        gotoxy(22,13);
        cprintf("  z katsayısı negatif. Bu bölgede kalan ");
        gotoxy(22,14);
        cprintf("  fonksiyonlar bu devre ile gerçekleştirilemezler. ");
        gotoxy(22,15);
        cprintf(" ");
        goto menu;
    }
    cprintf("
                                -1 2      \n\n");
    cprintf("                                % f(1-Z )      \n\n",n);
    cprintf("H%d(Z) = ----- \n\n",j);
    cprintf("                                -1      -2      \n\n");
    cprintf("                                1 %+fZ %+fZ      \n\n",p,r);

    if(text==0){
        fprintf(dosya,"
                                -1 2      \n");
        fprintf(dosya,"                                % f(1-Z )      \n",n);
        fprintf(dosya,"H%d(Z) = ----- \n",j);
        fprintf(dosya,"                                -1      -2      \n");
        fprintf(dosya,"                                1 %+fZ %+fZ      \n",p,r);
    }
}
/*NORMLIZASYON KATSAYISININ BULUNDUGU BOLUM*/
f=0;
k1=n*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc));
k2=n*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc));

```

```

k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc))) ;
k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-2][0];
T[j-1][0]=T1;
fx[sayac13]=f;
fy[sayac13]=T1;
sayac13=1;
max=T1;
for(f=1+((Fc)/400);f<=Fc/2;f+=((Fc)/400){
    k1=n*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc)) ;
    k2=n*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc)) ;
    k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc))) ;
    k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
    T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-2][sayac13];
    T[j-1][sayac13]=T1;
    fx[sayac13]=f;
    fy[sayac13]=T1;
    sayac13++;
    if(T1>max)
        max=T1;
}
f=0;
k5=n*(1+p+r)*(1-cos(2*PI*f/Fc)) ;
k6=n*(1+p+r)*sin(2*PI*f/Fc) ;
k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
T2=sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8))*T[j-2][0];
sayac13=1;
max1=T2;
for(f=1+((Fc)/400);f<=Fc/2;f+=Fc/400){
    k5=n*(1+p+r)*(1-cos(2*PI*f/Fc)) ;
    k6=n*(1+p+r)*sin(2*PI*f/Fc) ;
    k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
    k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
    T2=(sqrt((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)))*T[j-2][sayac13];
    sayac13++;
    if(T2>max1)
        max1=T2;
}
h[j]=max/max1;
/*      SONU      */

D[j-1][3]=(n)/r;
D[j-1][1]=1+(p)+r;
D[j-1][2]=1/h[j];
D[j-1][4]=(1/r)/h[j];
D[j-1][5]=1;
D[j-1][6]=(1/r)-1;

enkucuk=D[j-1][1];
for (sayac=2;sayac<3;sayac++)
    if(D[j-1][sayac]<enkucuk)
        enkucuk=D[j-1][sayac];
carpan1=min/enkucuk;
enkucuk=D[j-1][3];
for (sayac=4;sayac<7;sayac++)
    if(D[j-1][sayac]<enkucuk)
        enkucuk=D[j-1][sayac];
carpan2=min/enkucuk;

```

```

        C[j-1][2]=D[j-1][1]*carpan1;
        C[j-1][3]=D[j-1][2]*carpan1;
        C[j-1][1]=D[j-1][3]*carpan2;
        C[j-1][4]=D[j-1][4]*carpan2;
        C[j-1][5]=D[j-1][5]*carpan2;
        C[j-1][6]=D[j-1][6]*carpan2;

        for(g=0;g<10e5;g++)
            s+=g;
    }
    cprintf("\r\n");
    sayac4=0;
    for(sayac3=3;sayac3<1e12; sayac3=sayac3+3){
        for(sayac2=1 ;sayac2<7 ;sayac2++){
            for (sayac1=sayac4; sayac1<sayac3 ; sayac1++){
                if(sayac1<k){
                    cprintf("C[%d][%d]=%10.5fpf ",sayac1+1,sayac2,C[sayac1][sayac2]);
                    if(text==0){
                        fprintf(dosya,"C[%d][%d]=%10.5fpf ",sayac1,sayac2,C[sayac1][sayac2]);
                    }
                }
            }
            cprintf("\r\n");
            if(text==0){
                fprintf(dosya,"\n");
            }
        }
        cprintf("\r\n");
        if(text==0){
            fprintf(dosya,"\n");
        }
        sayac4=sayac4+3;
        if(sayac1>=k) goto menu;
    }
    fclose(dosya);
    for(j=0;j<30;j++)
        isim1[j]=0;
}
else {
    k=(N-1)/2;
    cprintf("                %d                \r\n",k);
    cprintf("H(Z)=Ho(Z)*TT Hj(Z) ; N=%d. DERECE HGS\r\n",N);
    cprintf("                j=1                \r\n");
    strcat(isim1,uz1);
    if((dosya=fopen(isim1,"wb"))==NULL){
        text=1;
    }

    q=Wcutoff/(2*Fc+Wcutoff);
    w=(2*Fc-Wcutoff)/(Wcutoff+2*Fc);
    cprintf("                -1                \r\n"),
    cprintf("                %f(1-Z )                \r\n",q*t);
    cprintf("Ho(Z) = ----- \r\n");
    cprintf("                -1\r\n");
    cprintf("                1%+fZ                \r\n",w);
    if(text==0){
        fprintf(dosya,"                %d                \n",k);
        fprintf(dosya,"H(Z)=Ho(Z)*TT Hj(Z) ; N=%d. DERECE HGS\n",N);
        fprintf(dosya,"                j=1                \n");
    }
}

```

```

fprintf(dosya,"
fprintf(dosya,"
fprintf(dosya,"Ho(Z) = ————— \n");
fprintf(dosya,"
fprintf(dosya,"
}
sayac13=0;
max=0;
for(f=0;f<=Fc/2;f+=(Fc)/400){
    k1=q*t*(1-cos(2*PI*f/Fc));
    k2=q*t*(sin(2*PI*f/Fc));
    k3=1+(w*cos(2*PI*f/Fc));
    k4=w*sin(2*PI*f/Fc);
    T1=sqrt(((k1*k1)+(k2*k2))/((k3*k3)+(k4*k4)));
    T[0][sayac13]=T1;
    fx[sayac13]=f;
    fy[sayac13]=T1;
    if(T1>max)
        max=T1;
    sayac13++;
}
C[0][2]=1;
C[0][3]=(1-w)/(1+w);
C[0][1]=(C[0][2]+C[0][3])*q*t;
enkucuk=C[0][1];
for (sayac=1;sayac<4;sayac++)
    if(C[0][sayac]<enkucuk)
        enkucuk=C[0][sayac];
carpan1=min/enkucuk;
C[0][3]=C[0][3]*carpan1;
C[0][2]=C[0][2]*carpan1;
C[0][1]=C[0][1]*carpan1;
C[0][4]=0;
C[0][5]=0;
C[0][6]=0;
for(j=1;j<=k;j++){
    Bk=2*sin(((2*j-1)*PI)/(2*N));
    m=((4*Fc*Fc)+(2*Fc*Bk*Wcutoff)+(Wcutoff*Wcutoff));
    n=(Wcutoff*Wcutoff)/m;
    p=((8*Fc*Fc)-(2*Wcutoff*Wcutoff))/m;
    r=((4*Fc*Fc)-(2*Bk*Fc*Wcutoff)+(Wcutoff*Wcutoff))/m;
    if(r<0){
        window(22,10,66,15);
        textbackground(4);
        gotoxy(22,11);
        cprintf(" ");
        gotoxy(22,12);
        cprintf(" || -2 || ");
        gotoxy(22,13);
        cprintf(" || z katsayısı negatif. Bu bölgede kalan || ");
        gotoxy(22,14);
        cprintf(" || fonksiyonlar bu devre ile gerçekleştirilemezler. || ");
        gotoxy(22,15);
        cprintf(" ");
        goto menu;
    }
}
cprintf("
cprintf("
cprintf("H%d(Z) = ————— \n",j);

```

```

cprintf("          -1      -2      \r\n ");
cprintf("          1 %fZ %fZ      \r\n",p,r);
if(text==0){
fprintf(dosya,"          -1 2      \n");
fprintf(dosya,"          % f(1-Z )      \n",n);
fprintf(dosya,"H%d(Z) = ----- \n",j);
fprintf(dosya,"          -1      -2      \n");
fprintf(dosya,"          1 %fZ %fZ      \n",p,r);
}
/*NORMLIZASYON KATSAYISININ BULUNDUGU BOLUM*/

f=0;
k1=n*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc)) ;
k2=n*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc)) ;
k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc))) ;
k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-1][0];
T[j][0]=T1;
fx[sayac13]=f;
fy[sayac13]=T1;
sayac13=1;
max=T1;
for(f=1+((Fc)/400);f<=Fc/2;f+=(Fc)/400){
    k1=n*(1-(2*cos(2*PI*f/Fc))+cos(2*2*PI*f/Fc)) ;
    k2=n*(2*sin(2*PI*f/Fc)-sin(2*2*PI*f/Fc)) ;
    k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc))) ;
    k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
    T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[j-1][sayac13];
    T[j][sayac13]=T1;
    fx[sayac13]=f;
    fy[sayac13]=T1;
    sayac13++;
    if(T1>max)
        max=T1;
}
f=0;
k5=n*(1+p+r)*(1-cos(2*PI*f/Fc)) ;
k6=n*(1+p+r)*sin(2*PI*f/Fc) ;
k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
T2=sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8))*T[j-1][0];
sayac13=1;
max1=T2;
for(f=1+((Fc)/400);f<=Fc/2;f+=(Fc)/400){
    k5=n*(1+p+r)*(1-cos(2*PI*f/Fc)) ;
    k6=n*(1+p+r)*sin(2*PI*f/Fc) ;
    k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
    k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
    T2=(sqrt((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)))*T[j-1][sayac13];
    sayac13++;
    if(T2>max1)
        max1=T2;
}
h[j]=max/max1;
/*          SONU          */

D[j][3]=(n)/r;
D[j][1]=1+(p)+r;
D[j][2]=1/h[j];

```

```

D[j][4]=(1/r)/h[j];
D[j][5]=1;
D[j][6]=(1/r)-1;

enkucuk=D[j][1];
for (sayac=2;sayac<3;sayac++)
    if(D[j][sayac]<enkucuk)
        enkucuk=D[j][sayac];
carpan1=min/enkucuk;
enkucuk=D[j][3];
for (sayac=4;sayac<7;sayac++)
    if(D[j][sayac]<enkucuk)
        enkucuk=D[j][sayac];
carpan2=min/enkucuk;
C[j][2]=D[j][1]*carpan1;
C[j][3]=D[j][2]*carpan1;
C[j][1]=D[j][3]*carpan2;
C[j][4]=D[j][4]*carpan2;
C[j][5]=D[j][5]*carpan2;
C[j][6]=D[j][6]*carpan2;

for(g=0;g<10e5;g++)
    s+=g;
}
cprintf("\r\n");
sayac4=0;
for(sayac3=3;sayac3<1e12; sayac3=sayac3+3){
    for(sayac2=1 ;sayac2<7 ;sayac2++){
        for (sayac1=sayac4; sayac1<sayac3 ; sayac1++){
            if(sayac1<k+1){
                cprintf("C[%d][%d]=%10.5fpf ",sayac1,sayac2,C [sayac1][sayac2]);
                if(text==0){
                    fprintf(dosya,"C[%d][%d]=%10.5fpf ",sayac1, sayac2,C[sayac1][sayac2]);
                }
            }
        }
    }
    cprintf("\r\n");
    if(text==0){
        fprintf(dosya,"\n");
    }
}
cprintf("\r\n");
if(text==0){
    fprintf(dosya,"\n");
}
sayac4=sayac4+3;
if(sayac1>=k+1) goto menu;
}
fclose(dosya);
for(j=0;j<30;j++)
    isim1[j]=0;
}
menu:
do{
    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=0;
            BITTI2=1;

```

```

        window(1,1,80,25);
        textbackground(1);
        clrscr();
        break;
    case F10 :
        BITTI1=1;
        BITTI2=1;
        window(1,1,80,25);
        textbackground(1);
        clrscr();
        break;
    case F1 :
        gettext(1,1,80,25,sakla);
        spawnle_devreh();
        _setcursortype(_NOCURS);
        puttext(1,1,80,25,sakla);
        break;
    case F2 :
        sayac13=0;
        eksenx_max=Fc/2;
        if((Fc/2)<=10 && (Fc/2)>0)
            eksenx_max1=10;
        else if((Fc/2)<=100 && (Fc/2)>10)
            eksenx_max1=100;
        else if((Fc/2)<=1000 && (Fc/2)>100)
            eksenx_max1=1000;
        else if((Fc/2)<=10000 && (Fc/2)>1000)
            eksenx_max1=10000;
        else if((Fc/2)<=100000 && (Fc/2)>10000)
            eksenx_max1=100000;
        else if((Fc/2)<=1000000 && (Fc/2)>100000)
            eksenx_max1=1000000;
        else
            eksenx_max1=Fc/2;
        ekseny_max=max;
        for(sayac13=0;sayac13<200;sayac13++){
            if(N%2==0)
                ekseny[sayac13]=T[k-1][sayac13];
            else
                ekseny[sayac13]=T[k][sayac13];
        }
        gettext(1,1,80,25,sakla);
        grafik();
        _setcursortype(_NOCURS);
        puttext(1,1,80,25,sakla);
        break;
    }
} while(!BITTI2);
exit :
} while(!BITTI1);
}

void directory()
{
    struct find_t fblk;
    int i,j,done,bitti,filecount;
    char ad[2000][30],b[]=".";
    for(j=0;j<30;j++)
        isim[j]=0;

```

```

window(1,1,80,25);
textbackground(1);
if(ara==0)
done = _dos_findfirst("*.ags",_A_NORMAL,&ffblk);
else if(ara==1)
done = _dos_findfirst("*.ygs",_A_NORMAL,&ffblk);
i=0;
while(!done){
    strcpy(ad[i],ffblk.name);
    i++;
    done = _dos_findnext(&ffblk);
}
filecount=i;
_setcursortype(_NOCURSOR);
gotoxy(10,16);printf(" ");
gotoxy(10,17);printf(" ");
gotoxy(10,18);printf(" ");
gotoxy(10,19);printf(" ");
gotoxy(10,20);printf(" ");
gotoxy(10,21);printf(" ");
gotoxy(10,22);printf(" ");
gotoxy(10,23);printf(" ");
gotoxy(10,24);printf(" ");
gotoxy(10,25);printf(" ");
window(11,17,29,24);
bitti=0;
gotoxy(1,1);
if(filecount==0)
    goto git;
else if(filecount<7){
    for(j=0;j<filecount;j++)
        cprintf(" %-18s",ad[j]);
}
else{
    for(j=0;j<7;j++)
        cprintf(" %-18s",ad[j]);
}
textattr(15+((16)<<2));
gotoxy(1,1);
cprintf(" %-17s",ad[0]);
i=0;
do{
    switch(KAR=tusal()){

    case ASAGI_TUSU :

        if(i!=filecount-1){

            i++;
            gotoxy(1,wherey());
            textattr(23+((24)<<5));
            textcolor(15);
            cprintf(" %-18s",ad[i-1]);
            gotoxy(1,wherey());
            textattr(15+((16)<<2));
            cprintf(" %-17s",ad[i]);
            break;
        }
        else {

```

```

        break;
    }
case YUKARI_TUSU :
    if(i==0)break;
    if(wherex()==1){
        gotoxy(1,1);
        textattr(15+((16)<<2));
        cprintf(" %-17s",ad[i-1]);
        textattr(23+((24)<<5));
        textcolor(15);
        gotoxy(1,2);cprintf(" %-17s",ad[i]);
        gotoxy(1,3);cprintf(" %-17s",ad[i+1]);
        gotoxy(1,4);cprintf(" %-17s",ad[i+2]);
        gotoxy(1,5);cprintf(" %-17s",ad[i+3]);
        gotoxy(1,6);cprintf(" %-17s",ad[i+4]);
        gotoxy(1,7);cprintf(" %-17s",ad[i+5]);
        gotoxy(1,8);cprintf(" %-17s",ad[i+6]);
        gotoxy(1,1);
        i--;
        break;
    }
    else{
        gotoxy(1,wherex());
        textattr(23+((24)<<5));
        textcolor(15);
        cprintf(" %-17s",ad[i]);
        gotoxy(1,wherex()-1);
        textattr(15+((16)<<2));
        cprintf(" %-17s",ad[i-1]);
        i--;
        break;
    }
case CR :
    strcpy(isim,ad[i]);
    strtok(isim,b);
    bitti=1;
    break;

case ESC :
    don :
        bitti=1;
        break;
    default :
        bitti=0;
        break;
}
}while(!bitti);
git :
window(10,16,60,25);
textbackground(1);
textattr(23+((24)<<5));
textcolor(15);
clrscr();
_setcursortype(_NORMALCURSOR);
}

```

BGS.CPP

```

#include<stdio.h>
#include<graphics.h>
#include<process.h>
#include<complex.h>
#include<math.h>
#include<stdlib.h>
#include<dos.h>
#include<bios.h>
#include<conio.h>
#include<fcntl.h>
#include<string.h>
#include<sys\stat.h>
#include "cursor1.h"
#define PI (float) (3.141592654)
#define CR 13
#define ESC 27
#define BACK_SPACE 8
#define HOME 327
#define END 335
#define YUKARI_TUSU 328
#define ASAGI_TUSU 336
#define SOL_TUSU 331
#define SAG_TUSU 333
#define PAGE_UP 329
#define PAGE_DOWN 337
#define F1 315 /*AGS*/
#define F2 316 /*HGS*/
#define F3 317 /*BGS*/
#define F4 318 /*YARDIM */
#define F5 319 /*ÇIKIŞ*/
#define F6 320
#define F7 321 /*YÜKLEME*/
#define F8 322 /*SAKLAMA*/
#define F9 323 /*HESAPLAMA*/
#define F10 324 /*ANA MENÜYE DÖNÜŞ*/
#define F11 325
#define F12 326
#define DEL 339 /*SATIR SİLME*/
void directory();
void grafik();
FILE *dosya;
int KAR,sayfa,sayfa1,ara;
float eksenx_max,eksenx_max1,eksenx,/*ekseny[410],*/ekseny_max,fx[410],fy[410];
char BITTI,BITTI1,BITTI2,BITTI3,BITTI4,BITTI5;
char METIN[200],isim[30],sakla[20000];
int satir;
fpos_t POZIS;
int u_x(float u)
{
    return((int)(log10(u)*570*(1/log10(eksenx_max1))));
}
int v_y(float v)
{
    return((int)(390+(v*325)*(120/((ceil(ekseny_max+0.2))*100))));
}
void UKS_line (float u1, float v1, float u2,float v2)
{

```

```

line ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_outtextxy(float u1,float v1)
{
    outtextxy(u_x(u1),v_y(v1),".");
}
void UKS_circle (float u, float v, float r)
{
    circle ( u_x(u), v_y(v), u_x(r) );
}
void grafik()
{
    int GrAdapter,GrMod;
    int hata_kodu,bol,sayac1=0,sayac2;
    float ymax,boly;
    int ilkcurposx,ilkcurposy;
    int soncurposx,soncurposy;
    int boyut,bitti6=0,l,belirle=0;
    int ESCTusu,eksy[410],eksx[410];
    char katar[31];
    detectgraph ( &GrAdapter,&GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter , &GrMod ,"" );
    hata_kodu = graphresult();
    setviewport(0,0,639,479,0);
    if(hata_kodu != grOk)
    {
        printf("grafik hatası : (#%d kodlu hata)! %s\n",hata_kodu,grapherrormsg(hata_kodu));
        getch();
        exit(1);
    }
    setlinestyle(3,10,1);
    line(60,9,637,9);
    line(67,9,67,401);
    line(67,401,637,401);
    line(637,9,637,401);
    line(60,87,67,87);
    line(60,166,67,166);
    line(60,244,67,244);
    line(60,323,67,323);
    line(60,401,67,401);
    ymax=ceil(ekseny_max+0.2);
    boly=(float)ymax/5;
    sprintf(katar,"%f",ymax);
    outtextxy(30,7,katar);
    sprintf(katar,"%f",ymax-boly);
    outtextxy(30,85,katar);
    sprintf(katar,"%f",ymax-(2*boly));
    outtextxy(30,164,katar);
    sprintf(katar,"%f",ymax-(3*boly));
    outtextxy(30,242,katar);
    sprintf(katar,"%f",ymax-(4*boly));
    outtextxy(30,321,katar);
    outtextxy(40,399,"0");
    setlinestyle(0,1,1);
    setttextjustify(1,2);
    if(eksenx_max1==100000){
        bol=570/5 ;
        line(67,401,67,405);
    }
}

```

```

    outtextxy(67,410,"1h");
    line(67+bol,401,181,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol,401,295,405);
    outtextxy(67+2*bol,410,"100h");
    line(67+3*bol,401,409,405);
    outtextxy(67+3*bol,410,"1Kh");
    line(67+4*bol,401,523,405);
    outtextxy(67+4*bol,410,"10Kh");
    line(67+5*bol,401,637,405);
    outtextxy(49+5*bol,410,"100Kh");
}
else if(eksenx_max1==10000){
    bol=int(570/4);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+bol,401,67+bol,405);
    outtextxy(67+bol,410,"10h");
    line(67+2*bol+1,401,67+2*bol+1,405);
    outtextxy(67+2*bol+1,410,"100h");
    line(67+3*bol+1,401,67+3*bol+1,405);
    outtextxy(67+3*bol+1,410,"1Kh");
    line(67+4*bol+2,401,67+4*bol+2,405);
    outtextxy(53+4*bol+2,410,"10Kh");
}
else if(eksenx_max1==1000){
    bol=int(570/6);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+91,401,67+91,405);
    outtextxy(67+91,410,"3h");
    line(67+2*bol,401,67+2*bol,405);
    outtextxy(67+2*bol,410,"10h");
    line(67+2*bol+91,401,67+2*bol+91,405);
    outtextxy(67+2*bol+91,410,"30h");
    line(67+4*bol,401,67+4*bol,405);
    outtextxy(67+4*bol,410,"100h");
    line(67+4*bol+91,401,67+4*bol+91,405);
    outtextxy(67+4*bol+91,410,"300h");
    line(67+6*bol,401,67+6*bol,405);
    outtextxy(57+6*bol,410,"1Kh");
}
else if(eksenx_max1==100){
    bol=int(570/4);
    line(67,401,67,405);
    outtextxy(67,410,"1h");
    line(67+136,401,67+136,405);
    outtextxy(67+136,410,"3h");
    line(67+2*bol+1,401,67+2*bol+1,405);
    outtextxy(67+2*bol+1,410,"10h");
    line(67+2*bol+136+1,401,67+2*bol+136+1,405);
    outtextxy(67+2*bol+136+1,410,"30h");
    line(67+4*bol+2,401,67+4*bol+2,405);
    outtextxy(53+4*bol+2,410,"100h");
}
else if(eksenx_max1==10){
    bol=int(570/2);
    line(67,401,67,405);
    outtextxy(67,410,"1h");

```

```

        line(67+272,401,67+272,405);
        outtextxy(67+272,410,"3h");
        line(637,401,637,405);
        outtextxy(627,410,"10h");
    }
    settxtjustify(0,2);
    outtextxy(310,451,"FREKANS");
    setviewport(1,469,200,479,1);
    setcolor(63);
    setfillstyle(SOLID_FILL,getcolor());
    bar(0,0,60,10);
    setcolor(0);
    outtextxy(10,1,"ÇIKIŞ ");
    setcolor(63);
    outtextxy(80,1,"KURSÖR");
    setlinestyle(0,1,1);
    setviewport(67,10,637,400,1);
    setbkcolor( 0 );
    setcolor(58);
    for(eksenx=1;eksenx<eksenx_max;eksenx+=((eksenx_max-1)/400)){

        eksx[sayac1]=u_x(eksenx);
        if(sayac1!=409){
            UKS_line(eksenx,-fy[sayac1],eksenx+((eksenx_max-1)/400),-fy[sayac1+1]);
        }
        sayac1++;
    }
    setcolor(63);
    do
    {
        switch(getch())
        {
            case 77:
                setviewport(1,469,200,479,1);
                clearviewport();
                setcolor(63);
                setfillstyle(SOLID_FILL,getcolor());
                bar(70,0,130,10);
                setcolor(0);
                outtextxy(80,1,"KURSÖR");
                setcolor(63);
                outtextxy(10,1,"ÇIKIŞ");
                setviewport(67,10,637,400,1);
                setbkcolor( 0 );
                belirle=1;
                break;

            case 75 :
                setviewport(1,469,200,479,1);
                clearviewport();
                setcolor(63);
                setfillstyle(SOLID_FILL,getcolor());
                bar(0,0,60,10);
                setcolor(0);
                outtextxy(10,1,"ÇIKIŞ");
                setcolor(63);
                outtextxy(80,1,"KURSÖR");
                setviewport(67,10,637,400,1);
                setbkcolor( 0 );
                belirle=0;

```

```

        break;
case CR:
    if(belirle==1){
        setviewport(1,469,200,479,1);
        clearviewport();
        setcolor(63);
        setfillstyle(SOLID_FILL,getcolor());
        bar(0,0,60,10);
        setcolor(0);
        outtextxy(10,1,"ÇIKIŞ");
        setcolor(63);
        setviewport(67,10,637,400,1);
        setbkcolor( 0 );
        ilkcurposx=0;
        ilkcurposy=v_y(-fy[0]);
        for(l=0;l<400;l++){
            eksy[l]=v_y(-fy[l]);
        }
        boyut=getmaxx();
        ESCTusu=grcursor(fx, fy,ilkcurposx,ilkcurposy,boyut,1,eksy,eksx,
            u_x(eksenx_max),ENTER_KEY,&soncurposx,&soncurposy);
        setviewport(1,469,200,479,1);
        clearviewport();
        setcolor(63);
        setfillstyle(SOLID_FILL,getcolor());
        bar(0,0,60,10);
        setcolor(0);
        outtextxy(10,1,"ÇIKIŞ");
        setcolor(63);
        outtextxy(80,1,"KURSÖR");
        setviewport(67,10,637,400,1);
        setbkcolor( 0 );
        belirle=0;
        ilkcurposx=soncurposx;
        ilkcurposy=soncurposy;
        setviewport(450,440,639,479,0);
        clearviewport();
        setviewport(67,10,637,400,1);
        bitti6=0;
        }
        else{
            bitti6=1;
        }
        break;
    }
}while(!bitti6);

closegraph();
}
int tusal(void)
{
    int TUS,lo, hi;
    TUS=bioskey(0);
    lo=TUS & 0X00FF;
    hi=(TUS & 0XFF00) >> 8;
    return((lo==0) ? hi+256 : lo);
}

void spawnle_devreb(void)

```

```

{
    int result;
    result = spawnle(P_WAIT, "devreb.exe", NULL, NULL);
    if (result == -1)
    {
        perror("Error from spawnle");
        exit(1);
    }
}

void main()
{
    char kr,boyut[2],a1[30],isim1[30],uz[]=".bgs",uz1[]=".bdt";
    do{
        float C[10][8],D[10][7],E[2][6],z[2],T[10][410],h[20],F[11],Y[10],G[6][4],p1,q1,DISK1;
        float DISK2,DISK3,DISK4,n1,A1,B1,B2,enkucuk,carpan1,carpan2,min=0,f,s,bul,bul1,bul2;
        float bul3,bul4,bul5;
        float Fcl=0,Fch=0,Fc=0,B,Wo,Fctl,Fcth,x,y,m,n,p,r,l,t,Qp,Kv=0,k1,k2,k3,k4,T1,T2,max;
        float max1,k5,k6,k7,k8,b1 ;
        int text=0,N=0,b2,sayac,i,sayac1,sayac2,sayac3,sayac4,k,j,sayac5,sayac6,sayac7,kr1,sayac8;
        int sayac9,sayac10,sayac13;
        BITTI=0;
        BITTI1=0;
        BITTI2=0;
        clrscr();
        _setcursortype(_NORMALCURSOR);
        gotoxy(10,5) ;printf("
        gotoxy(10,6) ;printf("
        gotoxy(10,7) ;printf("
        gotoxy(10,8) ;printf("
        gotoxy(10,9) ;printf("
        gotoxy(10,10) ;printf("
        gotoxy(10,11) ;printf("
        gotoxy(10,12) ;printf("
        gotoxy(10,13) ;printf("
        gotoxy(10,14) ;printf("
        window(62,1,79,4);
        gotoxy(62,2);cprintf("
        gotoxy(62,3);cprintf("
        gotoxy(62,4);cprintf("
        window(62,4,79,7);
        gotoxy(62,5);cprintf("
        gotoxy(62,6);cprintf("
        gotoxy(62,7);cprintf("
        window(62,7,79,10);
        gotoxy(62,8) ;cprintf("
        gotoxy(62,9) ;cprintf("
        gotoxy(62,10);cprintf("
        window(62,10,79,13);
        gotoxy(62,11);cprintf("
        gotoxy(62,12);cprintf("
        gotoxy(62,13);cprintf("
        window(62,13,79,16);
        gotoxy(62,14);cprintf("
        gotoxy(62,15);cprintf("
        gotoxy(62,16);cprintf("
        window(1,1,61,25);
        gotoxy(20,8);
        sayac8=0;
    }
}

```

LÜTFEN DİJİTAL ÖZELLİKLERİ GİRİNİZ

Fcl(Hz)	=	
Fch(Hz)	=	
Fc(Hz)	=	
Kv(dB)	=	
N	=	
MKD(pf)	=	

F7 YÜKLEME

F8 SAKLAMA

F9 HESAPLAMA

F10 ANA MENÜ

DEL SATIR SİLME

```

do{
    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=1;
            goto exit;

        case BACK_SPACE :
            if(wherex()==20)
                break;
            printf("\b");
            sayac9=wherex();
            clreol();
            gotoxy(45,wherey());printf(" || ");
            gotoxy(sayac9,wherey());
            a1[sayac8-1]=0;
            sayac8--;
            b1=atof(a1);
            break;

        case CR :
            if(wherex()==20){
                printf("\a");
                break;
            }
            else{
                if(wherey()==8) {
                    b1=atof(a1);
                    Fcl=b1;
                }
                else if(wherey()==9){
                    b1=atof(a1);
                    Fch=b1;
                }
                else if(wherey()==10){
                    b1=atof(a1);
                    Fc=b1;
                }
                else if(wherey()==11) {
                    b1=atof(a1);
                    Kv=b1;
                }
                else if(wherey()==12) {
                    b2=atoi(a1);
                    N=b2;
                }
                else if(wherey()==13) {
                    b1=atof(a1);
                    min=b1;
                }
                for(j=0;j<sayac8+1;j++)
                    a1[j]=0;
                sayac8=0;
                if(wherey()<=13)
                    gotoxy(20,wherey()+1);
                if(wherey()==14)
                    gotoxy(20,8);
                break;
            }
    }
default :

```

```

if(KAR>=48 && KAR<=57 || KAR==46){

    a1[sayac8]=KAR;
    sayac8++;
    gotoxy(wherex(),wherey());printf("%c",KAR);
    break;
}
else{

    printf("\a");
    break;
}
}
case YUKARI_TUSU :

    if(wherey()==8)
        break;
    else
        gotoxy(wherex(),wherey()-1);
        break;
case ASAGI_TUSU :

    if(wherey()==13)
        break;
    else
        gotoxy(wherex(),wherey()+1);
        break;
case DEL :

    if(wherey()==8){
        gotoxy(20,8);
        clreol();
        gotoxy(45,8);printf(" || ");
        gotoxy(20,8);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherey()==9){
        gotoxy(20,9);
        clreol();
        gotoxy(45,9);printf(" || ");
        gotoxy(20,9);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
    else if(wherey()==10){
        gotoxy(20,10);
        clreol();
        gotoxy(45,10);printf(" || ");
        gotoxy(20,10);
        for(j=0;j<sayac8+1;j++)
            a1[j]=0;

        sayac8=0;
        break;
    }
}

```

```

else if(wherely()==11){
    gotoxy(20,11);
    clreol();
    gotoxy(45,11);printf(" || ");
    gotoxy(20,11);
    for(j=0;j<sayac8+1;j++)
        a1[j]=0;
    sayac8=0;
    break;
}
else if(wherely()==12){
    gotoxy(20,12);
    clreol();
    gotoxy(45,12);printf(" || ");
    gotoxy(20,12);
    for(j=0;j<sayac8+1;j++)
        a1[j]=0;
    sayac8=0;
    break;
}
else if(wherely()==13) {
    gotoxy(20,13);
    clreol();
    gotoxy(45,13);printf(" || ");
    gotoxy(20,13);
    for(j=0;j<sayac8+1;j++)
        a1[j]=0;
    sayac8=0;
    break;
}
}

case F7 :

    for(j=0;j<30;j++)
        isim[j]=0;

yuk2:
BITTI4=0;
sayac8=0;
window(62,16,79,19);
gotoxy(62,17);cprintf(" ");
gotoxy(62,18);cprintf(" F6 DOSYA LİSTESİ ");
gotoxy(62,19);cprintf(" ");
window(1,1,61,25);
gotoxy(10,2);printf(" ");
gotoxy(10,3);printf(" DOSYA İSMİ: ");
gotoxy(10,4);printf(" ");
gotoxy(22,3);
do{
switch(KAR=tusal())
{
case F7 :
    window(62,16,79,19);
    clrscr();
    window(1,1,61,25);
    gotoxy(10,2);clreol();
    gotoxy(10,3);clreol();
    gotoxy(10,4);clreol();
    gotoxy(20,8);

```

```

        BITTI4=1;
        break;
case F6 :

        directory();
        window(1,1,61,25);
        gotoxy(22,3);
        goto g;
default :
        if(KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
            KAR>=97 && KAR<=122){
                isim[sayac8]=KAR;
                sayac8++;
                printf("%c",KAR);
                break;
        }
        else{
                printf("\a");
                break;
        }
case BACK_SPACE :
        if(wherex()==22)
                break;
        printf("\b");
        sayac9=wherex();
        clreol();
        gotoxy(45,wherey());printf(" || ");
        gotoxy(sayac9,wherey());
        isim[sayac8-1]=0;
        sayac8--;
        break;
case CR :
g :
        window(62,16,79,19);
        clrscr();
        window(1,1,61,25);
        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        strcpy(isim1,isim);
        strcat(isim,uz);
        if((dosya=fopen(isim,"rb"))==NULL){
                for(j=0;j<sayac8+5;j++)
                        isim[j]=0;
                sayac8=0;
                goto yuk2;
        }
        fscanf(dosya,"%f\n%f\n%f\n%f\n%d\n%f\n",&Fcl,&Fch,&Fc,&Kv,&N,&min);
        gotoxy(20,8);clreol();
        gotoxy(45,8);printf(" || ");
        gotoxy(20,8) ;printf("%f",Fcl);
        gotoxy(20,9);clreol();
        gotoxy(45,9);printf(" || ");
        gotoxy(20,9) ;printf("%f",Fch);
        gotoxy(20,10);clreol();
        gotoxy(45,10);printf(" || ");
        gotoxy(20,10);printf("%f",Fc);
        gotoxy(20,11);clreol();
        gotoxy(45,11);printf(" || ");

```

```

gotoxy(20,11);printf("%f",Kv);
gotoxy(20,12);clreol();
gotoxy(45,12);printf("||");
gotoxy(20,12);printf("%d",N);
gotoxy(20,13);clreol();
gotoxy(45,13);printf("||");
gotoxy(20,13);printf("%f",min );
gotoxy(20,8);
fclose(dosya);
for(j=0;j<sayac8+5;j++)
    isim[j]=0;

BITTI4=1;
break;
}
}while(!BITTI4);
break;
case F8 :
    for(j=0;j<30;j++)
        isim[j]=0;

    BITTI5=0;
    sayac8=0;
    sak2 :
    gotoxy(10,2);printf(" ");
    gotoxy(10,3);printf("DOSYA İSMİ:");
    gotoxy(10,4);printf(" ");
    gotoxy(22,3);
    do{
        switch(KAR=tusal())
        {
            case F8 :
                gotoxy(10,2);clreol();
                gotoxy(10,3);clreol();
                gotoxy(10,4);clreol();
                BITTI5=1;
                break;

            default :
                if((KAR>=48 && KAR<=57 || KAR>=65 && KAR<=90 ||
                    KAR>=97 && KAR<=122)){
                    isim[sayac8]=KAR;
                    sayac8++;
                    printf("%c",KAR);
                    break;
                }
                else{
                    printf("\a");
                    break;
                }
        }
    }
    case BACK_SPACE :
        if(wherex()==22)
            break;
        printf("\b");
        sayac9=wherex();
        clreol();
        gotoxy(45,wherey());printf("||");
        gotoxy(sayac9,wherey());
        isim[sayac8-1]=0;
        sayac8--;
        break;

    case CR :

```

```

        gotoxy(10,2);clreol();
        gotoxy(10,3);clreol();
        gotoxy(10,4);clreol();
        strcpy(isim1,isim);
        strcat(isim,uz);
        gotoxy(20,8);
        if((dosya=fopen(isim,"wb"))==NULL){
            for(j=0;j<sayac8+5;j++)
                isim[j]=0;

            sayac8=0;
            goto sak2;
        }
        fprintf(dosya,"%f\n%f\n%f\n%f\n%d\n%f\n",Fcl,Fch,Fc,Kv,N,min);
        fclose(dosya);
        for(j=0;j<sayac8+5;j++)
            isim[j]=0;

        BITTI5=1;
        break;
    }
}while(!BITTI5);
gotoxy(20,8);
break;

case F10 :
    BITTI1=1;
    goto exit;

case F9 :
    BITTI=1;
    break;
}
}
while(!BITTI);

window(1,1,80,25);
textbackground(0);
clrscr();
window(1,1,80,24);
textbackground(1);
clrscr();
window(1,24,80,25);
textbackground(4);
cprintf(" ESC PAR MENÜ");
cprintf("    F1 DEVRE YAPISI");
cprintf("    F2 FREKANS EĞRİS");
cprintf("    F10 ANA MENÜ ");
window(1,1,80,23);
textbackground(1);
clrscr();
_setcursortype(_NOCURS);
if(Fcl>=Fch || min<=0){
    window(22,10,57,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("
    goto menu;
}

x=(PI*Fch)/Fc;

```

GİRİLEN KARAKTERİSTİKLER BAND
 GEÇİREN FİLTREYİ GERÇEKLEMEZ.

```

y=(PI*Fcl)/Fc;
Fcth=(Fc/PI)*tan(x);
Fctl=(Fc/PI)*tan(y);
B=2*PI*(Fcth-Fctl);
Wo=2*PI*sqrt(Fcth*Fctl);
m=((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
n=(2*B*Fc)/m;
p=((2*Wo*Wo)-(8*Fc*Fc))/m;
r=((4*Fc*Fc)-(2*B*Fc)+(Wo*Wo))/m;
l=Kv/20;
t=pow(10,l);
if(N==2){
    if(r<0){
        window(22,10,66,15);
        textbackground(4);
        gotoxy(22,11);
        cprintf("||");
        gotoxy(22,12);
        cprintf("|| -2 ||");
        gotoxy(22,13);
        cprintf("|| z katsayısı negatif. Bu bölgede kalan ||");
        gotoxy(22,14);
        cprintf("|| fonksiyonlar bu devre ile gerçekleştirilemezler. ||");
        gotoxy(22,15);
        cprintf("||");
        goto menu;
    }

    Qp=sqrt(Fch*Fcl)/(Fch-Fcl);
    strcat(isim1,uz1);
    if((dosya=fopen(isim1,"wb"))==NULL){
        text=1;
    }

    cprintf("          -1  -1      \r\n");
    cprintf("          % f(1-Z) (1+Z)      \r\n",t*n);
    cprintf("H(Z) =-----\r\n");
    cprintf("          -1  -2      \r\n");
    cprintf("          1 %+fZ %+fZ      \r\n",p,r);

    if(text==0){
        fprintf(dosya,"          -1  -1      \n");
        fprintf(dosya,"          % f(1-Z) (1+Z)      \n",t*n);
        fprintf(dosya,"H(Z) =-----\n");
        fprintf(dosya,"          -1  -2      \n");
        fprintf(dosya,"          1 %+fZ %+fZ      \n",p,r);
    }
}
/*NORMLİZASYON KATSAYISININ BULUNDUĞU BÖLÜM*/

f=1;
k1=t*n*(1-cos(2*2*PI*f/Fc));
k2=t*n*sin(2*2*PI*f/Fc);
k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc)));
k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
max=T1;
sayac13=0;
for(f=1;f<=Fc/2;f+=(Fc/2)-1)/400){
    k1=t*n*(1-cos(2*2*PI*f/Fc));

```

```

k2=t*n*sin(2*2*PI*f/Fc) ;
k3=(1+(r*cos(2*2*PI*f/Fc))+(p*cos(2*PI*f/Fc))) ;
k4=(r*sin(2*2*PI*f/Fc))+(p*sin(2*PI*f/Fc));
T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
T[0][sayac13]=T1;
fy[sayac13]=T1;
fx[sayac13]=f;
sayac13++;
if(T1>max)
    max=T1;
}

f=1;
k5=t*n*(1+p+r)*(1+cos(2*PI*f/Fc)) ;
k6=t*n*(1+p+r)*sin(2*PI*f/Fc) ;
k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
T2=sqrt(((k5*k5)+(k6*k6)))/((k7*k7)+(k8*k8));
max1=T2;
for(f=1;f<=Fc/2;f+=(Fc/2)-1)/400){
    k5=t*n*(1+p+r)*(1+cos(2*PI*f/Fc)) ;
    k6=t*n*(1+p+r)*sin(2*PI*f/Fc) ;
    k7=(1+(p*cos(2*PI*f/Fc))+(r*cos(2*2*PI*f/Fc)));
    k8=((p*sin(2*PI*f/Fc))+(r*sin(2*2*PI*f/Fc)));
    T2=(sqrt((k5*k5)+(k6*k6)))/(sqrt((k7*k7)+(k8*k8)));
    if(T2>max1)
        max1=T2;
}
h[1]=max/max1;
/*      SONU      */
D[0][0]=1+(p)+r;
D[0][1]=1/h[1];
D[0][2]=(1/r)/h[1];
D[0][3]=1;
D[0][4]=(1/r)-1;
D[0][5]=(t*n)/r;
D[0][6]=(t*n)/r;

enkucuk=D[0][0];
for (sayac=1;sayac<2;sayac++)
    if(D[0][sayac]<enkucuk)
        enkucuk=D[0][sayac];
carpan1=min/enkucuk;
enkucuk=D[0][2];
for (sayac=3;sayac<7;sayac++)
    if(D[0][sayac]<enkucuk)
        enkucuk=D[0][sayac];
carpan2=min/enkucuk;
C[0][3]=D[0][0]*carpan1;
C[0][4]=D[0][1]*carpan1;
C[0][5]=D[0][2]*carpan2;
C[0][6]=D[0][3]*carpan2;
C[0][7]=D[0][4]*carpan2;
C[0][1]=D[0][5]*carpan2;
C[0][2]=D[0][6]*carpan2;
cprintf("\r\n");
cprintf("C[1][1]=%10.5fpf\r\n",C[0][1]);
cprintf("C[1][2]=%10.5fpf\r\n",C[0][2]);
cprintf("C[1][3]=%10.5fpf\r\n",C[0][3]);

```

```

cprintf("C[1][4]=%10.5fpf\r\n",C[0][4]);
cprintf("C[1][5]=%10.5fpf\r\n",C[0][5]);
cprintf("C[1][6]=%10.5fpf\r\n",C[0][6]);
cprintf("C[1][7]=%10.5fpf\r\n",C[0][7]);
if(text==0){
fprintf(dosya,"C[1][1]=%10.5fpf\r\n",C[0][1]);
fprintf(dosya,"C[1][2]=%10.5fpf\r\n",C[0][2]);
fprintf(dosya,"C[1][3]=%10.5fpf\r\n",C[0][3]);
fprintf(dosya,"C[1][4]=%10.5fpf\r\n",C[0][4]);
fprintf(dosya,"C[1][5]=%10.5fpf\r\n",C[0][5]);
fprintf(dosya,"C[1][6]=%10.5fpf\r\n",C[0][6]);
fprintf(dosya,"C[1][7]=%10.5fpf\r\n",C[0][7]);
}
}
if(N>10){
window(22,10,66,15);
textbackground(4);
gotoxy(22,11);cprintf("
gotoxy(22,12);cprintf("
gotoxy(22,13);cprintf("
gotoxy(22,14);cprintf("
goto menu;
}
if(N>2 && N<=10){
window(30,10,51,13);
textbackground(4);
gotoxy(30,11);cprintf("
gotoxy(30,12);cprintf("
gotoxy(30,13);cprintf("
if(N==4){
F[0]=(16*pow(Fc,4)+sqrt(2)*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)+
(B*B)*4*pow(Fc,2)+sqrt(2)*B*2*Fc*Wo*Wo+pow(Wo,4));
F[1]=(-64*pow(Fc,4)-16*B*sqrt(2)*pow(Fc,3)+4*sqrt(2)*B*Fc*Wo*Wo+
4*pow(Wo,4))/F[0];
F[2]=(96*pow(Fc,4)-8*Fc*Fc*(2*Wo*Wo+B*B)+6*pow(Wo,4))/F[0];
F[3]=(-64*pow(Fc,4)+16*sqrt(2)*B*pow(Fc,3)-4*sqrt(2)*B*Fc*Wo*Wo+
4*pow(Wo,4))/F[0];
F[4]=(16*pow(Fc,4)-sqrt(2)*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)+
(B*B)*4*pow(Fc,2)-sqrt(2)*B*2*Fc*Wo*Wo+pow(Wo,4))/F[0];
F[5]=-(4*B*B*Fc*Fc)/F[0];
/*İTERASYON*/
bul1=0.1;
for(G[1][1]=2 ; G[1][1]>-2 ; G[1][1]=-0.000001){

if((((F[1]-G[1][1])*G[1][1])-F[2])*(((F[1]-G[1][1])*G[1][1])-F[2])-4*F[4])<=0)
cprintf("");
else if((F[3]*F[3]-4*(F[1]-G[1][1])*F[4]*G[1][1])<=0)
cprintf("");

else {
DISK1=-((((F[1]-G[1][1])*G[1][1])-F[2])+sqrt((((F[1]-G[1][1])*
G[1][1])-F[2])*(((F[1]-G[1][1])* G[1][1])-F[2])-4*F[4]))/2;
DISK2=-((((F[1]-G[1][1])*G[1][1])-F[2])-sqrt((((F[1]-G[1][1])*G[1][1])-F[2])
*((F[1]-G[1][1])*G[1][1])-F[2])-4*F[4]))/2;
DISK3=(F[3]+sqrt(F[3]*F[3]-4*(F[1]-G[1][1])*F[4]*G[1][1]))/(2*(F[1]-G[1][1]));
DISK4=(F[3]-sqrt(F[3]*F[3]-4*(F[1]-G[1][1])*F[4]*G[1][1]))/(2*(F[1]-G[1][1]));
if(fabs(DISK1)<1 && fabs(DISK3)<1 && fabs(DISK1-DISK3)<0.1){
if(fabs(DISK1-DISK3)<bul1){

```

```

        bul1=fabs(DISK1-DISK3);
        G[1][2]=DISK1;
        bul=G[1][1];
    }
}
if(fabs(DISK1)<1 && fabs(DISK4)<1 && fabs(DISK1-DISK4)<0.1){
    if(fabs(DISK1-DISK4)<bul1){
        bul1=fabs(DISK1-DISK4);
        G[1][2]=DISK1;
        bul=G[1][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK3)<1 && fabs(DISK2-DISK3)<0.1){
    if(fabs(DISK2-DISK3)<bul1){
        bul1=fabs(DISK2-DISK3);
        G[1][2]=DISK2;
        bul=G[1][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK4)<1 && fabs(DISK2-DISK4)<0.1){
    if(fabs(DISK2-DISK4)<bul1){
        bul1=fabs(DISK2-DISK4);
        G[1][2]=DISK2;
        bul=G[1][1];
    }
}
else
    cprintf("");
}
}
G[1][1]=bul;
/*İTERASYON*/
/*NORMLİZASYON KATSAYISININ BULUNDUĞU BÖLÜM*/
G[0][2]=F[4]/G[1][2];
G[0][1]=F[1]-G[1][1];
if(F[5]<0){
    G[0][3]=-sqrt(fabs(F[5]));
    G[1][3]=-G[0][3];
}
else{
    G[0][3]=sqrt(F[5]);
    G[1][3]=G[0][3];
}
/*if(G[1][2]<0 || G[0][2]<0){
    window(22,10,66,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("
    gotoxy(22,15);cprintf("
    goto menu;
}*/
window(1,1,80,23);
textbackground(1);
clrscr();
cprintf("
cprintf("
cprintf("H(Z) = -----");

```

-2
z katsayısı negatif. Bu bölgede kalan
fonksiyonlar bu devre ile gerçekleştirilemezler.

```

        -1 2   -1 2           \r\n");
        % f(1-Z ) (1+Z )      \r\n",F[5]);

```

```

cprintf("          -1      -2      -3      -4\n ");
cprintf("          1 %+fZ %+fZ %+fZ %+fZ      \n",F[1],F[2],F[3],F[4]);
cprintf("\n");
strcat(isim1,uz1);
if((dosya=fopen(isim1,"wb"))==NULL){
    text=1;
}

if(text==0){
    fprintf(dosya,"          -1 2      -1 2      \n");
    fprintf(dosya,"          %f(1-Z ) (1+Z )      \n",F[5]);
    fprintf(dosya,"H(Z)=-----\n");
    fprintf(dosya,"          -1      -2      -3      -4\n ");
    fprintf(dosya,"          1 %+fZ %+fZ %+fZ %+fZ      \n",F[1],F[2],F[3],F[4]);
}
}

if(N==6){
Y[0]=(16*pow(Fc,4)+B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
      +(B*B)*4*pow(Fc,2)+B*2*Fc*Wo*Wo+pow(Wo,4));
Y[1]=(-64*pow(Fc,4)-16*B*pow(Fc,3)+4*B*Fc*Wo*Wo+4*pow(Wo,4))/Y[0];
Y[2]=(96*pow(Fc,4)-8*Fc*Fc*(2*Wo*Wo+B*B)+6*pow(Wo,4))/Y[0];
Y[3]=(-64*pow(Fc,4)+16*B*pow(Fc,3)-4*B*Fc*Wo*Wo+4*pow(Wo,4))/Y[0];
Y[4]=(16*pow(Fc,4)-B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
      +(B*B)*4*pow(Fc,2)-B*2*Fc*Wo*Wo+pow(Wo,4))/Y[0];
Y[5]=(4*B*B*Fc*Fc)/Y[0];
G[0][1]=((2*Wo*Wo)-(8*Fc*Fc))/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
G[0][2]=((4*Fc*Fc)-(2*B*Fc)+(Wo*Wo))/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
G[0][3]=((2*B*Fc))/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
F[0]=Y[1]+G[0][1];
F[1]=Y[2]+(G[0][1]*Y[1])+G[0][2];
F[2]=Y[3]+(G[0][1]*Y[2])+G[0][2]*Y[1];
F[3]=Y[4]+(G[0][1]*Y[3])+G[0][2]*Y[2];
F[4]=(G[0][1]*Y[4])+G[0][2]*Y[3];
F[5]=G[0][2]*Y[4];
F[6]=G[0][3]*Y[5];
bul1=0.1;
/*ITERASYON*/
for(G[2][1]=2; G[2][1]>-2; G[2][1]=0.000001){
    if((((Y[1]-G[2][1])*G[2][1])-Y[2])*(((Y[1]-G[2][1])*G[2][1])-Y[2])-4*Y[4])<=0)
        cprintf("");
    else if((Y[3]*Y[3]-4*(Y[1]-G[2][1])*Y[4]*G[2][1])<=0)
        cprintf("");
    else {
        DISK1=-((((Y[1]-G[2][1])*G[2][1])-Y[2])+sqrt((((Y[1]-G[2][1])*G[2][1])-Y[2])
            * (((Y[1]-G[2][1])*G[2][1])-Y[2])-4*Y[4]))/2;
        DISK2=-((((Y[1]-G[2][1])*G[2][1])-Y[2])-sqrt((((Y[1]-G[2][1])*G[2][1])-Y[2])
            * ((Y[1]-G[2][1])*G[2][1]-Y[2])-4*Y[4]))/2;
        DISK3=(Y[3]+sqrt(Y[3]*Y[3]-4*(Y[1]-G[2][1])*Y[4]*G[2][1]))/(2*(Y[1]
            -G[2][1]));
        DISK4=(Y[3]-sqrt(Y[3]*Y[3]-4*(Y[1]-G[2][1])*Y[4]*G[2][1]))/(2*(Y[1]
            -G[2][1]));
        if(fabs(DISK1)<1 && fabs(DISK3)<1 && fabs(DISK1-DISK3)<0.1){
            if(fabs(DISK1-DISK3)<bul1){
                bul1=fabs(DISK1-DISK3);
                G[2][2]=DISK1;
                bul=G[2][1];
            }
        }
        if(fabs(DISK1)<1 && fabs(DISK4)<1 && fabs(DISK1-DISK4)<0.1){
            if(fabs(DISK1-DISK4)<bul1){

```

```

        bul1=fabs(DISK1-DISK4);
        G[2][2]=DISK1;
        bul=G[2][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK3)<1 && fabs(DISK2-DISK3)<0.1){
    if(fabs(DISK2-DISK3)<bul1){
        bul1=fabs(DISK2-DISK3);
        G[2][2]=DISK2;
        bul=G[2][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK4)<1 && fabs(DISK2-DISK4)<0.1){
    if(fabs(DISK2-DISK4)<bul1){
        bul1=fabs(DISK2-DISK4);
        G[2][2]=DISK2;
        bul=G[2][1];
    }
}

else
    cprintf("");
}
}
/*İTERASYON*/
/*NORMLİZASYON KATSAYISININ BULUNDUĞU BÖLÜM*/
G[2][1]=bul;
G[1][2]=Y[4]/G[2][2];
G[1][1]=Y[1]-G[2][1];
if(F[5]<0){
    G[1][3]=-sqrt(fabs(Y[5]));
    G[2][3]=-G[1][3];
}
else{
    G[1][3]=sqrt(Y[5]);
    G[2][3]=G[1][3];
}
}
/*if(G[1][2]<0 || G[0][2]<0 || G[2][2]<0){
    window(22,10,66,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("
    gotoxy(22,15);cprintf("
    goto menu;
}*/
window(1,1,80,23);
textbackground(1);
clrscr();
cprintf("
cprintf("
cprintf("H(Z) = -----");
cprintf("
cprintf("
cprintf("
F[0],F[1], F[2],F[3],F[4],F[5]);

strcat(isim1,uz1);
if((dosya=fopen(isim1,"wb"))==NULL){
    text=1;
}

```

```

if(text==0){
fprintf(dosya,"
                                -1 3   -1 3           \n");
fprintf(dosya,"
                                % f(1-Z ) (1+Z )           \n",F[6]);
fprintf(dosya,"H(Z) = ----- \n");
fprintf(dosya,"
                                -1      -2      -3      -4      -5      -6\n ");
fprintf(dosya,"
                                1 %+fZ %+fZ %+fZ %+fZ %+fZ %+fZ           \n",
                                F[0], F[1],F[2],F[3],F[4],F[5]);
}
/*SONU*/
}
if(N==8){
z[0]=0.7654;
z[1]=1.84776;
for(j=0;j<2;j++){
E[j][0]=(16*pow(Fc,4)+z[j]*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
          +(B*B)*4*pow(Fc,2)+z[j]*B*2*Fc*Wo*Wo+pow(Wo,4));
E[j][1]=(-64*pow(Fc,4)-16*B*z[j]*pow(Fc,3)+4*z[j]*B*Fc*Wo*Wo
          +4*pow(Wo,4))/E[j][0];
E[j][2]=(96*pow(Fc,4)-8*Fc*Fc*(2*Wo*Wo+B*B)+6*pow(Wo,4))/E[j][0];
E[j][3]=(-64*pow(Fc,4)+16*z[j]*B*pow(Fc,3)-4*z[j]*B*Fc*Wo*Wo
          +4*pow(Wo,4))/E[j][0];
E[j][4]=(16*pow(Fc,4)-z[j]*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
          +(B*B)*4*pow(Fc,2)-z[j]*B*2*Fc*Wo*Wo+pow(Wo,4))/E[j][0];
E[j][5]=(4*B*B*Fc*Fc)/E[j][0];
/*ITERASYON*/
bul1=0.1;
for(G[2*j+1][1]=2 ; G[2*j+1][1]>-2 ; G[2*j+1][1]=-0.000001){

if((((E[j][1]-G[2*j+1][1])*G[2*j+1][1]-E[j][2])*((E[j][1]-G[2*j+1][1])
          *G[2*j+1][1]-E[j][2])-4*E[j][4])<=0)
          cprintf("");
else if((E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+1][1])*E[j][4]*G[2*j+1][1])<=0)
          cprintf("");
else {
DISK1=-((((E[j][1]-G[2*j+1][1])*G[2*j+1][1]-E[j][2])+sqrt((((E[j][1]-
G[2*j+1][1])*G[2*j+1][1]-E[j][2])*((E[j][1]-G[2*j+1][1])*G[2*j+1][1]-E[j][2])-4*E[j][4]))/2;
DISK2=-((((E[j][1]-G[2*j+1][1])*G[2*j+1][1]-E[j][2])-sqrt((((E[j][1]-
G[2*j+1][1])*G[2*j+1][1]-E[j][2])*((E[j][1]-G[2*j+1][1])*G[2*j+1][1]-E[j][2])-4*E[j][4]))/2;
DISK3=(E[j][3]+sqrt(E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+1][1])*E[j][4]
          *G[2*j+1][1]))/(2*(E[j][1]-G[2*j+1][1]));
DISK4=(E[j][3]-sqrt(E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+1][1])*E[j][4]
          *G[2*j+1][1]))/(2*(E[j][1]-G[2*j+1][1]));
if(fabs(DISK1)<1 && fabs(DISK3)<1 && fabs(DISK1-DISK3)<0.1){
          if(fabs(DISK1-DISK3)<bul1){
bul1=fabs(DISK1-DISK3);
G[2*j+1][2]=DISK1;
bul=G[2*j+1][1];
          }
          }
if(fabs(DISK1)<1 && fabs(DISK4)<1 && fabs(DISK1-DISK4)<0.1){
          if(fabs(DISK1-DISK4)<bul1){
bul1=fabs(DISK1-DISK4);
G[2*j+1][2]=DISK1;
bul=G[2*j+1][1];
          }
          }
if(fabs(DISK2)<1 && fabs(DISK3)<1 && fabs(DISK2-DISK3)<0.1){
          if(fabs(DISK2-DISK3)<bul1){
bul1=fabs(DISK2-DISK3);

```

```
G[2*j+1][2]=DISK2;
bul=G[2*j+1][1];
}
}
if(fabs(DISK2)<1 && fabs(DISK4)<1 && fabs(DISK2-DISK4)<0.1){
    if(fabs(DISK2-DISK4)<bul){
        bul1=fabs(DISK2-DISK4);
        G[2*j+1][2]=DISK2;
        bul=G[2*j+1][1];
    }
}

else
    cprintf("");
}
}
G[2*j+1][1]=bul;
/*İTERASYON*/
/*NORMLİZASYON KATSAYISININ BULUNDUĞU BÖLÜM*/
G[2*j][2]=E[j][4]/G[2*j+1][2];
G[2*j][1]=E[j][1]-G[2*j+1][1];

if(E[j][5]<0){
    G[2*j][3]=-sqrt(fabs(E[j][5]));
    G[2*j+1][3]=-G[2*j][3];
}
else{
    G[2*j][3]=sqrt(E[j][5]);
    G[2*j+1][3]=G[2*j][3];
}
}
G[0][3]=G[0][3]*t;
F[0]=E[0][5]*E[1][5];
F[1]=E[1][1]+E[0][1];
F[2]=E[1][2]+(E[0][1]*E[1][1])+E[0][2];
F[3]=E[1][3]+(E[0][1]*E[1][2])+(E[0][2]*E[1][1])+E[0][3];
F[4]=E[1][4]+(E[0][1]*E[1][3])+(E[0][2]*E[1][2])+(E[0][3]*E[1][1])+E[0][4];
F[5]=(E[0][1]*E[1][4])+(E[0][2]*E[1][3])+(E[0][3]*E[1][2])+(E[0][4]*E[1][1]);
F[6]=(E[0][2]*E[1][4])+(E[0][3]*E[1][3])+(E[0][4]*E[1][2]);
F[7]=(E[0][3]*E[1][4])+(E[0][4]*E[1][3]);
F[8]=E[0][4]*E[1][4];
/*if(G[1][2]<0 || G[0][2]<0 || G[2][2]<0 || G[3][2]<0){
    window(22,10,66,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
-2
z katsayısı negatif. Bu bölgede kalan
fonksiyonlar bu devre ile gerçekleştirilemezler.");
    gotoxy(22,12);cprintf("
");
    gotoxy(22,13);cprintf("
");
    gotoxy(22,14);cprintf("
");
    gotoxy(22,15);cprintf("
");
    goto menu;
}
*/
window(1,1,80,23);
textbackground(1);
clrscr();
cprintf("
-1 4 -1 4\r\n");
cprintf("
% f(1-Z ) (1+Z ) \r\n",t*F[0]);
cprintf("H(Z)=-----");
cprintf("
-1 -2 -3 -4 -5 -6 -7 -8\r\n");
cprintf("
1%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%
```

```

\rfn",F[1],F[2],F[3],F[4],F[5],F[6],F[7],F[8]);
strcat(isim1,uz1);
if((dosya=fopen(isim1,"wb"))==NULL){
    text=1;
}
if(text==0){
    fprintf(dosya,"
    -1 4    -1 4    \n");
    fprintf(dosya,"
    % f(1-Z ) (1+Z )    \n",t*F[0]);
    fprintf(dosya,"H(Z) = ----- \n");
    fprintf(dosya,"
    -1 -2 -3 -4 -5 -6 -7 -8 \n ");
    fprintf(dosya,"
    1%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%+.3fZ%
    \n ",F[1],F[2],F[3],F[4],F[5],F[6],F[7],F[8]);
}
}
if(N==10){
z[0]=0.618;
z[1]=1.618;
G[0][1]=((2*Wo*Wo)-(8*Fc*Fc))/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
G[0][2]=((4*Fc*Fc)-(2*B*Fc)+(Wo*Wo))/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
G[0][3]=t*(2*B*Fc)/((4*Fc*Fc)+(2*B*Fc)+(Wo*Wo));
for(j=0;j<2;j++){
E[j][0]=(16*pow(Fc,4)+z[j]*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
+(B*B)*4*pow(Fc,2)+z[j]*B*2*Fc*Wo*Wo+pow(Wo,4));
E[j][1]=(-64*pow(Fc,4)-16*B*z[j]*pow(Fc,3)+4*z[j]*B*Fc*Wo*Wo
+4*pow(Wo,4))/E[j][0];
E[j][2]=(96*pow(Fc,4)-8*Fc*Fc*(2*Wo*Wo+B*B)+6*pow(Wo,4))/E[j][0];
E[j][3]=(-64*pow(Fc,4)+16*z[j]*B*pow(Fc,3)-4*z[j]*B*Fc*Wo*Wo
+4*pow(Wo,4))/E[j][0];
E[j][4]=(16*pow(Fc,4)-z[j]*B*8*pow(Fc,3)+(2*Wo*Wo)*4*pow(Fc,2)
+(B*B)*4*pow(Fc,2)-z[j]*B*2*Fc*Wo*Wo+pow(Wo,4))/E[j][0];
E[j][5]=(4*B*B*Fc*Fc)/E[j][0];
/*:ITERASYON*/
bul1=0.1;
for(G[2*j+2][1]=2 ; G[2*j+2][1]>-2 ; G[2*j+2][1]=0.000001){
    if((((E[j][1]-G[2*j+2][1])*G[2*j+2][1])-E[j][2])*((E[j][1]-G[2*j+2][1])
    *G[2*j+2][1]-E[j][2])-4*E[j][4])<=0)
        cprintf("");
    else if((E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+2][1])*E[j][4]*G[2*j+2][1])<=0)
        cprintf("");

    else {
        DISK1=-((((E[j][1]-G[2*j+2][1])*G[2*j+2][1])-E[j][2])+sqrt((((E[j][1]-
        G[2*j+2][1])*G[2*j+2][1]-E[j][2])*((E[j][1]-G[2*j+2][1])*G[2*j+2][1]-E[j][2])-4*E[j][4])))/2;
        DISK2=-((((E[j][1]-G[2*j+2][1])*G[2*j+2][1])-E[j][2])-sqrt((((E[j][1]-
        G[2*j+2][1])*G[2*j+2][1]-E[j][2])*((E[j][1]-G[2*j+2][1])*G[2*j+2][1]-E[j][2])-4*E[j][4])))/2;
        DISK3=(E[j][3]+sqrt(E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+2][1])*E[j][4]
        *G[2*j+2][1]))/(2*(E[j][1]-G[2*j+2][1]));
        DISK4=(E[j][3]-sqrt(E[j][3]*E[j][3]-4*(E[j][1]-G[2*j+2][1])*E[j][4]
        *G[2*j+2][1]))/(2*(E[j][1]-G[2*j+2][1]));
        if(fabs(DISK1)<1 && fabs(DISK3)<1 && fabs(DISK1-DISK3)<0.1){
            if(fabs(DISK1-DISK3)<bul1){
                bul1=fabs(DISK1-DISK3);
                G[2*j+2][2]=DISK1;
                bul=G[2*j+2][1];
            }
        }
        if(fabs(DISK1)<1 && fabs(DISK4)<1 && fabs(DISK1-DISK4)<0.1){
            if(fabs(DISK1-DISK4)<bul1){

```

```

        bul1=fabs(DISK1-DISK4);
        G[2*j+2][2]=DISK1;
        bul=G[2*j+2][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK3)<1 && fabs(DISK2-DISK3)<0.1){
    if(fabs(DISK2-DISK3)<bul1){
        bul1=fabs(DISK2-DISK3);
        G[2*j+2][2]=DISK2;
        bul=G[2*j+2][1];
    }
}
if(fabs(DISK2)<1 && fabs(DISK4)<1 && fabs(DISK2-DISK4)<0.1){
    if(fabs(DISK2-DISK4)<bul1){
        bul1=fabs(DISK2-DISK4);
        G[2*j+2][2]=DISK2;
        bul=G[2*j+2][1];
    }
}

else
    cprintf("");
}
}
G[2*j+2][1]=bul;
/*TERASYON*/
/*NORMLİZASYON KATSAYISININ BULUNDUĞU BÖLÜM*/
G[2*j+1][2]=E[j][4]/G[2*j+2][2];
G[2*j+1][1]=E[j][1]-G[2*j+2][1];
if(E[j][5]<0){
    G[2*j+2][3]=-sqrt(fabs(E[j][5]));
    G[2*j+1][3]=-G[2*j+2][3];
}
else{
    G[2*j+2][3]=sqrt(E[j][5]);
    G[2*j+1][3]=G[2*j+2][3];
}
}
Y[0]=E[0][5]*E[1][5];
Y[1]=E[1][1]+E[0][1];
Y[2]=E[1][2]+(E[0][1]*E[1][1])+E[0][2];
Y[3]=E[1][3]+(E[0][1]*E[1][2])+(E[0][2]*E[1][1])+E[0][3];
Y[4]=E[1][4]+(E[0][1]*E[1][3])+(E[0][2]*E[1][2])+(E[0][3]*E[1][1])+E[0][4];
Y[5]=(E[0][1]*E[1][4])+(E[0][2]*E[1][3])+(E[0][3]*E[1][2])+(E[0][4]*E[1][1]);
Y[6]=(E[0][2]*E[1][4])+(E[0][3]*E[1][3])+(E[0][4]*E[1][2]);
Y[7]=(E[0][3]*E[1][4])+(E[0][4]*E[1][3]);
Y[8]=E[0][4]*E[1][4];
F[0]=G[0][3]*E[0][5]*E[1][5];
F[1]=E[1][1]+E[0][1]+G[0][1];
F[2]=E[1][2]+(E[0][1]*E[1][1])+E[0][2]+G[0][2]+(G[0][1]*Y[1]);
F[3]=E[1][3]+(E[0][1]*E[1][2])+(E[0][2]*E[1][1])+E[0][3]+(G[0][2]*Y[1])
        +(G[0][1]*Y[2]);
F[4]=E[1][4]+(E[0][1]*E[1][3])+(E[0][2]*E[1][2])+(E[0][3]*E[1][1])+E[0][4]
        +(G[0][2]*Y[2])+(G[0][1]*Y[3]);
F[5]=(E[0][1]*E[1][4])+(E[0][2]*E[1][3])+(E[0][3]*E[1][2])+(E[0][4]*E[1][1])
        +(G[0][2]*Y[3])+(G[0][1]*Y[4]);
F[6]=(E[0][2]*E[1][4])+(E[0][3]*E[1][3])+(E[0][4]*E[1][2])+(G[0][2]*Y[4])
        +(G[0][1]*Y[5]);
F[7]=(E[0][3]*E[1][4])+(E[0][4]*E[1][3])+(G[0][2]*Y[5])+(G[0][1]*Y[6]);
F[8]=E[0][4]*E[1][4]+(G[0][2]*Y[6])+(G[0][1]*Y[7]);

```

```

F[9]=(G[0][2]*Y[7])+(G[0][1]*Y[8]);
F[10]=G[0][2]*Y[8];
/*if(G[1][2]<0 || G[0][2]<0 || G[2][2]<0 || G[3][2]<0 || G[4][2]<0){
    window(22,10,66,15);
    textbackground(4);
    gotoxy(22,11);cprintf("
    gotoxy(22,12);cprintf("
    gotoxy(22,13);cprintf("
    gotoxy(22,14);cprintf("
    gotoxy(22,15);cprintf("
    goto menu;
} */
window(1,1,80,23);
textbackground(1); clrscr();
cprintf("
cprintf("
cprintf("H(Z)=
cprintf("
cprintf("
cprintf("
strcat(isim1,uz1);
if((dosya=fopen(isim1,"wb"))==NULL){
    text=1;
}
if(text==0){
    fprintf(dosya,"
    fprintf(dosya,"
    fprintf(dosya,"H(Z) =
    fprintf(dosya,"
    fprintf(dosya,"
}
}
for(sayac10=0 ; sayac10<(N/2); sayac10++){
    f=1;
    k1=G[sayac10][3]*(1-cos(2*2*PI*f/Fc)) ;
    k2=G[sayac10][3]*sin(2*2*PI*f/Fc) ;
    k3=(1+(G[sayac10][2]*cos(2*2*PI*f/Fc))+(G[sayac10][1]*cos(2*PI*f/Fc))) ;
    k4=(G[sayac10][2]*sin(2*2*PI*f/Fc))+(G[sayac10][1]*sin(2*PI*f/Fc));
    if(sayac10==0)
        T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
    if(sayac10>0)
        T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)))*T[sayac10-1][0];
    T[sayac10][0]=T1;
    fy[0]=T1;
    fx[0]=0;
    sayac13=0;
    max=T1;
    for(f=1;f<=Fc/2;f+=((Fc/2)-1)/400){
        k1=G[sayac10][3]*(1-cos(2*2*PI*f/Fc)) ;
        k2=G[sayac10][3]*sin(2*2*PI*f/Fc) ;
        k3=(1+(G[sayac10][2] *cos(2*2*PI*f/Fc))+(G[sayac10][1]
            *cos(2*PI*f/Fc))) ;
        k4=(G[sayac10][2]*sin(2*2*PI*f/Fc))+(G[sayac10][1]*sin(2*PI*f/Fc));
        if(sayac10==0)
            T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4)));
        if(sayac10>0)
            T1=(sqrt((k1*k1)+(k2*k2)))/(sqrt((k3*k3)+(k4*k4))
                *T[sayac10-1][sayac13];

```

```

T[sayac10][sayac13]=T1;
fy[sayac13]=T1;
fx[sayac13]=f;
sayac13++;
if(T1>max)
    max=T1;
}
f=1;
k5=G[sayac10][3]*(1+G[sayac10][1]+G[sayac10][2])*(1+cos(2*PI*f/Fc));
k6=G[sayac10][3]*(1+G[sayac10][1]+G[sayac10][2])*sin(2*PI*f/Fc);
k7=(1+(G[sayac10][1]*cos(2*PI*f/Fc)))+(G[sayac10][2]*cos(2*PI*f/Fc));
k8=((G[sayac10][1]*sin(2*PI*f/Fc)))+(G[sayac10][2]*sin(2*PI*f/Fc));
if(sayac10==0)
    T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)));
if(sayac10>0)
    T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)))*T[sayac10-1][0];
max1=T2;
sayac13=0;
for(f=1;f<=Fc/2;f+=(Fc/2)-1/400){
    k5=G[sayac10][3]*(1+G[sayac10][1]+G[sayac10][2])*(1+cos(2*PI*f/Fc));
    k6=G[sayac10][3]*(1+G[sayac10][1]+G[sayac10][2])*sin(2*PI*f/Fc);
    k7=(1+(G[sayac10][1]*cos(2*PI*f/Fc)))+(G[sayac10][2]*cos(2*PI*f/Fc));
    k8=((G[sayac10][1]*sin(2*PI*f/Fc)))+(G[sayac10][2]*sin(2*PI*f/Fc));
    if(sayac10==0)
        T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)));
    if(sayac10>0)
        T2=sqrt(((k5*k5)+(k6*k6))/((k7*k7)+(k8*k8)))*T[sayac10-1][sayac13];
    sayac13++;
    if(T2>max1)
        max1=T2;
}
h[sayac10]=max/max1;

D[sayac10][0]=1+(G[sayac10][1]+G[sayac10][2]);
D[sayac10][1]=1/h[sayac10];
D[sayac10][2]=(1/G[sayac10][2])/h[sayac10];
D[sayac10][3]=1;
D[sayac10][4]=(1/G[sayac10][2])-1;
D[sayac10][5]=(G[sayac10][3])/G[sayac10][2];
D[sayac10][6]=(G[sayac10][3])/G[sayac10][2];

enkucuk=D[sayac10][0];
for (sayac=1;sayac<2;sayac++)
    if(D[sayac10][sayac]<enkucuk)
        enkucuk=D[sayac10][sayac];
carpan1=min/enkucuk;
enkucuk=D[sayac10][2];
for (sayac=3;sayac<7;sayac++)
    if(D[sayac10][sayac]<enkucuk)
        enkucuk=D[sayac10][sayac];
carpan2=min/enkucuk;
C[sayac10][3]=D[sayac10][0]*carpan1;
C[sayac10][4]=D[sayac10][1]*carpan1;
C[sayac10][5]=D[sayac10][2]*carpan2;
C[sayac10][6]=D[sayac10][3]*carpan2;
C[sayac10][7]=D[sayac10][4]*carpan2;
C[sayac10][1]=D[sayac10][5]*carpan2;

```

```

C[sayac10][2]=D[sayac10][6]*carpan2;
}
/*      SONU      */
fprintf(dosya,"\\n");
sayac4=0;
for(sayac3=3;sayac3<1e12; sayac3=sayac3+3){
    for(sayac2=1 ;sayac2<8 ;sayac2++){
        for (sayac1=sayac4; sayac1<sayac3 ; sayac1++){
            if(sayac1<N/2){
                fprintf("C[%d][%d]=%10.5fpf ",sayac1+1,sayac2,C[sayac1][sayac2]);
                if(text==0){
                    fprintf(dosya,"C[%d][%d]=%10.5fpf ",sayac1+1,sayac2,C[sayac1][sayac2]);
                }
            }
        }
        fprintf("\\r\\n");
        if(text==0){
            fprintf(dosya,"\\n");
        }
    }
    fprintf("\\r\\n");
    if(text==0){
        fprintf(dosya,"\\n");
    }
    sayac4=sayac4+3;
    if(sayac1>=N/2) goto menu;
}
}
fclose(dosya);
for(j=0;j<30;j++)
    isim1[j]=0;
menu :
do{
    switch(KAR=tusal())
    {
        case ESC :
            BITTI1=0;
            BITTI2=1;
            window(1,1,80,25);
            textbackground(1);
            clrscr();
            break;
        case F10 :
            BITTI1=1;
            BITTI2=1;
            window(1,1,80,25);
            textbackground(1);
            clrscr();
            break;
        case F1 :
            gettext(1,1,80,25,sakla);
            spawnle_devreb();
            _setcursortype(_NOCURS);
            puttext(1,1,80,25,sakla);
            break;
        case F2 :
            sayac13=0;

```

```

        eksenx_max=Fc/2;
        if((Fc/2)<=10 && (Fc/2)>0)
            eksenx_max1=10;
        else if((Fc/2)<=100 && (Fc/2)>10)
            eksenx_max1=100;
        else if((Fc/2)<=1000 && (Fc/2)>100)
            eksenx_max1=1000;
        else if((Fc/2)<=10000 && (Fc/2)>1000)
            eksenx_max1=10000;
        else if((Fc/2)<=100000 && (Fc/2)>10000)
            eksenx_max1=100000;
        else
            eksenx_max1=Fc/2;
        ekseny_max=max;
        gettext(1,1,80,25,sakla);
        grafik();
        _setcursortype(_NOCURS);
        puttext(1,1,80,25,sakla);
        break;
    }
    }while(!BITTI2);
exit :
    }while(!BITTI1);
}
void directory()
{
    struct find_t fblk;
    int i,j,done,bitti,filecount;
    char ad[2000][30],b[]=".";
    for(j=0;j<30;j++)
        isim[j]=0;
    window(1,1,80,25);
    textbackground(1);
    done = _dos_findfirst("*.bgs",_A_NORMAL,&fblk);
    i=0;
    while(!done){
        strcpy(ad[i],fblk.name);
        i++;
        done = _dos_findnext(&fblk);
    }
    filecount=i;
    _setcursortype(_NOCURS);
    gotoxy(10,16);printf("
    ");
    gotoxy(10,17);printf("
    ");
    gotoxy(10,18);printf("
    ");
    gotoxy(10,19);printf("
    ");
    gotoxy(10,20);printf("
    ");
    gotoxy(10,21);printf("
    ");
    gotoxy(10,22);printf("
    ");
    gotoxy(10,23);printf("
    ");
    gotoxy(10,24);printf("
    ");
    gotoxy(10,25);printf("
    ");
    window(11,17,29,24);
    bitti=0;
    gotoxy(1,1);
    if(filecount==0)
        goto git;
    else if(filecount<7){
        for(j=0;j<filecount;j++)

```

```

        cprintf(" %-18s",ad[j]);
    }
    else{
        for(j=0;j<7;j++)
            cprintf(" %-18s",ad[j]);
    }
    textattr(15+((16)<<2));
    gotoxy(1,1);
    cprintf(" %-17s",ad[0]);
    i=0;
    do{
        switch(KAR=tusal()){

        case ASAGI_TUSU :

            if(i!=filecount-1){

                i++;
                gotoxy(1,wherey());
                textattr(23+((24)<<5));
                textcolor(15);
                cprintf(" %-18s",ad[i-1]);
                gotoxy(1,wherey());
                textattr(15+((16)<<2));
                cprintf(" %-17s",ad[i]);
                break;
            }
            else {
                break;
            }
        case YUKARI_TUSU :
            if(i==0)break;
            if(wherey()==1){
                gotoxy(1,1);
                textattr(15+((16)<<2));
                cprintf(" %-17s",ad[i-1]);
                textattr(23+((24)<<5));
                textcolor(15);
                gotoxy(1,2);cprintf(" %-17s",ad[i]);
                gotoxy(1,3);cprintf(" %-17s",ad[i+1]);
                gotoxy(1,4);cprintf(" %-17s",ad[i+2]);
                gotoxy(1,5);cprintf(" %-17s",ad[i+3]);
                gotoxy(1,6);cprintf(" %-17s",ad[i+4]);
                gotoxy(1,7);cprintf(" %-17s",ad[i+5]);
                gotoxy(1,8);cprintf(" %-17s",ad[i+6]);
                gotoxy(1,1);
                i--;
                break;
            }
            else{

                gotoxy(1,wherey());
                textattr(23+((24)<<5));
                textcolor(15);
                cprintf(" %-17s",ad[i]);
                gotoxy(1,wherey()-1);
                textattr(15+((16)<<2));
                cprintf(" %-17s",ad[i-1]);
                i--;
            }
        }
    }

```

```

        break;
    }
    case CR :
        strcpy(isim,ad[i]);
        strtok(isim,b);
        bitti=1;
        break;

    case ESC :
        don :
            bitti=1;
            break;
        default :
            bitti=0;
            break;
    }
} while(!bitti);
git :
window(10,16,60,25);
textbackground(1);
textattr(23+((24)<< 5));
textcolor(15);
clrscr();
_setcursortype(_NORMALCURSOR);
}

```

DEVREC.CPP

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<bios.h>
#include<graphics.h>
#define ESC 27
int tusal(void)
{
    int TUS,lo, hi;
    TUS=bioskey(0);
    lo=TUS & 0X00FF;
    hi=(TUS & 0XFF00) >> 8;
    return((lo==0) ? hi+256 : lo);
}
int u_x(float u)
{
    return((int)(0.79875*u));
}
int v_y(float v)
{
    return((int)(0.59875*v));
}
void UKS_line (float u1, float v1, float u2,float v2)
{
    line ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_rectangle (float u1, float v1, float u2,float v2)
{
    rectangle ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}

```

```

void UKS_circle (float u, float v, float r)
{
    circle ( u_x(u), v_y(v), u_x(r) );
}
void UKS_outtextxy(float u, float v,char *s)
{
    outtextxy(u_x(u),v_y(v), s);
}
void gecikmeli_iletkenlik(float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1-25;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-75,v1,u2-75,v2);
    UKS_line(u1-100,v1,u2-75,v2+10);
    UKS_line(u1-110,v1+10,u2-65,v2+10);
    UKS_line(u1-108,v1+13,u2-67,v2+13);
    UKS_line(u1-106,v1+16,u2-69,v2+16);
    UKS_line(u1-50,v1+20,u2-25,v2+40);
    UKS_line(u1-60,v1+40,u2-15,v2+40);
    UKS_line(u1-60,v1+45,u2-15,v2+45);
    UKS_line(u1-50,v1+45,u2-25,v1+65);
    UKS_line(u1-50,v1+65,u2-10,v2+75);
    UKS_line(u1-50,v1+65,u2-40,v2+75);
    UKS_line(u1-65,v1+75,u2-25,v2+85);
    UKS_line(u1-50,v1+85,u2-10,v2+75);
    UKS_line(u1-50,v1+85,u2-25,v2+105);
    UKS_line(u1-60,v1+105,u2-15,v2+105);
    UKS_line(u1-58,v1+108,u2-17,v2+108);
    UKS_line(u1-56,v1+111,u2-19,v2+111);
    UKS_line(u1-50,v1+20,u2,v2);
    UKS_line(u1-25,v1,u2-3,v2+8);
    UKS_line(u1-25,v1,u2-9,v2+2);
    UKS_outtextxy(u1-53,v1+69,"+");
    UKS_outtextxy(u1-53,v1+75,"-");
    UKS_circle(u1-50,v1+20,3);
    UKS_circle(u1-50,v1+20,2);
    UKS_circle(u1-50,v1+20,1);
    UKS_circle(u1-25,v1,3);
    UKS_circle(u1-25,v1,2);
    UKS_circle(u1-25,v1,1);
    UKS_circle(u1-75,v1,3);
    UKS_circle(u1-75,v1,2);
    UKS_circle(u1-75,v1,1);
    UKS_outtextxy(u1-40,v1-10,"e");
    UKS_outtextxy(u1-70,v1-10,"o");
}
void iletkenlik(float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1-25;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-75,v1,u2-75,v2);
    UKS_line(u1-100,v1,u2-75,v2+10);
    UKS_line(u1-110,v1+10,u2-65,v2+10);
    UKS_line(u1-108,v1+13,u2-67,v2+13);
    UKS_line(u1-106,v1+16,u2-69,v2+16);

```

```

UKS_line(u1-50,v1+20,u2-25,v2+40);
UKS_line(u1-60,v1+40,u2-15,v2+40);
UKS_line(u1-60,v1+45,u2-15,v2+45);
UKS_line(u1-50,v1+45,u2-25,v2+65);
UKS_line(u1-60,v1+65,u2-15,v2+65);
UKS_line(u1-58,v1+68,u2-17,v2+68);
UKS_line(u1-56,v1+71,u2-19,v2+71);
UKS_line(u1-50,v1+20,u2,v2);
UKS_line(u1-25,v1,u2-3,v2+8);
UKS_line(u1-25,v1,u2-9,v2+2);
UKS_circle(u1-50,v1+20,3);
UKS_circle(u1-50,v1+20,2);
UKS_circle(u1-50,v1+20,1);
UKS_circle(u1-25,v1,3);
UKS_circle(u1-25,v1,2);
UKS_circle(u1-25,v1,1);
UKS_circle(u1-75,v1,3);
UKS_circle(u1-75,v1,2);
UKS_circle(u1-75,v1,1);
UKS_outtextxy(u1-40,v1-10,"e");
UKS_outtextxy(u1-70,v1-10,"o");
}
void kapasite(float u1,float v1)
{
    float u2,v2;
    u2=u1;
    v2=v1+50;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-10,v1+50,u2+10,v2);
    UKS_line(u1-10,v1+55,u2+10,v2+5);
    UKS_line(u1,v1+55,u2,v2+55);
    UKS_line(u1-10,v1+105,u2+10,v2+55);
    UKS_line(u1-8,v1+108,u2+8,v2+58);
    UKS_line(u1-6,v1+111,u2+6,v2+61);
}
void iletkenlik1 (float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1+30;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1+80,v1,u2+65,v2);
    UKS_line(u1+95,v1,u2+65,v2+20);
    UKS_line(u1+85,v1+20,u2+75,v2+20);
    UKS_line(u1+87,v1+23,u2+73,v2+23);
    UKS_line(u1+89,v1+26,u2+71,v2+26);
    UKS_line(u1+55,v1+20,u2+25,v2+40);
    UKS_line(u1+45,v1+40,u2+35,v2+40);
    UKS_line(u1+45,v1+45,u2+35,v2+45);
    UKS_line(u1+55,v1+45,u2+25,v2+65);
    UKS_line(u1+45,v1+65,u2+35,v2+65);
    UKS_line(u1+47,v1+68,u2+33,v2+68);
    UKS_line(u1+49,v1+71,u2+31,v2+71);
    UKS_line(u1+55,v1+20,u2,v2);
    UKS_line(u1+30,v1,u2+3,v2+8);
    UKS_line(u1+30,v1,u2+9,v2+2);

    UKS_circle(u1+30,v1,3);
    UKS_circle(u1+30,v1,2);

```

```

UKS_circle(u1+30,v1,1);
UKS_circle(u1+80,v1,3);
UKS_circle(u1+80,v1,2);
UKS_circle(u1+80,v1,1);
UKS_circle(u1+55,v1+20,3);
UKS_circle(u1+55,v1+20,2);
UKS_circle(u1+55,v1+20,1);
UKS_outtextxy(u1+35,v1-10,"e");
UKS_outtextxy(u1+67,v1-10,"o");
}
void dugum(float u1,float v1)
{
    UKS_circle(u1,v1,3);
    UKS_circle(u1,v1,2);
    UKS_circle(u1,v1,1);
}
void main()
{
    int GrAdapter,GrMod,bitti=0;
    int hata_kodu;
    detectgraph ( &GrAdapter,&GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter , &GrMod ,"c:\\borlandc\\bgi" );
    hata_kodu = graphresult();
    if(hata_kodu != grOk)
    {
        printf("grafik hatası : (%d kodlu hata)! %s\n",hata_kodu ,grapherrormsg(hata_kodu));
        getch();
        exit(1);
    }
    setbkcolor( 1 );
    UKS_rectangle(200,100,250,150);
    UKS_rectangle(200,300,250,350);
    UKS_rectangle(400,100,450,150);
    UKS_rectangle(400,300,450,350);

    UKS_outtextxy(200,80,"CCII-1");
    UKS_outtextxy(200,280,"CCII-2");
    UKS_outtextxy(400,80,"CCII-3");
    UKS_outtextxy(400,280,"CCII-4");
    UKS_line(250,125,285,125);
    UKS_line(250,310,400,310);
    UKS_line(285,125,285,325);
    UKS_line(450,125,485,125);
    UKS_line(485,125,485,325);
    UKS_outtextxy(60,105,"VIN");
    dugum(100,110);
    UKS_line(600,325,620,325);
    UKS_outtextxy(630,320,"VOUT");
    dugum(620,325);
    UKS_outtextxy(205,110,"Y");
    UKS_outtextxy(205,135,"X");
    UKS_outtextxy(240,125,"Z");
    UKS_outtextxy(222,125,"-");
    UKS_outtextxy(205,310,"Y");
    UKS_outtextxy(205,335,"X");
    UKS_outtextxy(240,310,"Z");
    UKS_outtextxy(222,325,"+");
    UKS_outtextxy(405,110,"Y");

```

```

UKS_outtextxy(405,135,"X");
UKS_outtextxy(440,125,"Z");
UKS_outtextxy(422,125,"+");
UKS_outtextxy(405,310,"Y");
UKS_outtextxy(405,335,"X");
UKS_outtextxy(440,325,"Z");
UKS_outtextxy(422,325,"-");
UKS_outtextxy(347,409,"+");
UKS_outtextxy(347,415,"-");
UKS_outtextxy(370,412,"Vx4");
UKS_outtextxy(300,375,"Ci3");
UKS_outtextxy(390,180,"Ci4");
UKS_outtextxy(500,375,"Ci6");
UKS_line(450,325,600,325);
UKS_line(600,325,600,460);
UKS_line(600,460,50,460);
UKS_line(150,110,150,40);
UKS_line(150,40,375,40);
UKS_line(375,40,375,110);
UKS_line(375,110,400,110);
UKS_line(375,140,400,140);
UKS_line(50,460,50,310);
UKS_line(50,310,200,310);
UKS_line(200,110,100,110);
UKS_outtextxy(555,380,"Ci7");
UKS_outtextxy(165,180,"Ci1");
UKS_outtextxy(165,380,"Ci2");
UKS_outtextxy(365,380,"Ci5");
iletkenlik(200,140);
iletkenlik(200,340);
kapasite(375,140);
kapasite(285,325);
kapasite(485,325);
gecikmeli_iletkenlik(400,340);
iletkenlik1(485,340);
dugum(285,310);
dugum(485,325);
/*****/
UKS_rectangle(300,550,350,600);
UKS_line(300,560,260,560);
UKS_line(300,590,150,590);
UKS_line(350,575,375,575);
UKS_line(150,590,150,611);
UKS_line(280,590,280,611);
UKS_outtextxy(220,555,"VIN");
dugum(260,560);
UKS_outtextxy(305,560,"Y");
UKS_outtextxy(305,585,"X");
UKS_outtextxy(340,575,"Z");
UKS_outtextxy(322,575,"+");
UKS_line(375,575,460,575);
UKS_outtextxy(470,570,"VOUT");
dugum(460,575);
gecikmeli_iletkenlik(150,611);
UKS_outtextxy(115,651,"C01");
UKS_outtextxy(120,681,"Vx");
iletkenlik(280,611);
UKS_outtextxy(245,651,"C01");
UKS_line(450,575,450,596);

```

```

kapasite(450,596);
dugum(450,621);
UKS_outtextxy(465,646,"C02");
iletkenlik1(450,621);
gecikmeli_iletkenlik(450,621);
UKS_outtextxy(520,655,"C03");
UKS_outtextxy(360,661,"C03");
UKS_outtextxy(365,691,"Vz");
UKS_outtextxy(560,120,"2.DERECE CCII-SC ALÇAK");
UKS_outtextxy(570,140,"GEÇİREN AKTİF FİLTRE");
UKS_outtextxy(610,160,"BİRİM YAPISI");
UKS_outtextxy(610,180,"~1,2,3,...N");
UKS_outtextxy(560,580,"1.DERECE CCII-SC ALÇAK");
UKS_outtextxy(570,600,"GEÇİREN AKTİF FİLTRE");
UKS_outtextxy(610,620,"BİRİM YAPISI");
do{
switch(tusal())
{
case ESC :
        bitti=1;
        break;

default :
        printf("\a");
        break;
}
}while(!bitti);
closegraph();
}

```

DEVREH.CPP

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<bios.h>
#include<graphics.h>
#define ESC 27
int tusal(void)
{
    int TUS,lo, hi;
    TUS=bioskey(0);
    lo=TUS & 0X00FF;
    hi=(TUS & 0XFF00) >> 8;
    return((lo==0) ? hi+256 : lo);
}
int u_x(float u)
{
    return((int)(0.79875*u));
}
int v_y(float v)
{
    return((int)(0.59875*v));
}
void UKS_line (float u1, float v1, float u2,float v2)
{
    line ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_circle (float u, float v, float r)
{

```

```

circle ( u_x(u), v_y(v), u_x(r) );
}
void UKS_rectangle (float u1, float v1, float u2,float v2)
{
rectangle ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_outtextxy(float u, float v,char *s)
{
outtextxy(u_x(u),v_y(v), s);
}
void gecikmeli_iletkenlik(float u1,float v1)
{
float u2,v2;
v2=v1;
u2=u1-25;
UKS_line(u1,v1,u2,v2);
UKS_line(u1-75,v1,u2-75,v2);
UKS_line(u1-100,v1,u2-75,v2+10);
UKS_line(u1-110,v1+10,u2-65,v2+10);
UKS_line(u1-108,v1+13,u2-67,v2+13);
UKS_line(u1-106,v1+16,u2-69,v2+16);
UKS_line(u1-50,v1+20,u2-25,v2+40);
UKS_line(u1-60,v1+40,u2-15,v2+40);
UKS_line(u1-60,v1+45,u2-15,v2+45);
UKS_line(u1-50,v1+45,u2-25,v1+65);
UKS_line(u1-50,v1+65,u2-10,v2+75);
UKS_line(u1-50,v1+65,u2-40,v2+75);
UKS_line(u1-65,v1+75,u2-25,v2+85);
UKS_line(u1-50,v1+85,u2-10,v2+75);
UKS_line(u1-50,v1+85,u2-25,v2+105);
UKS_line(u1-60,v1+105,u2-15,v2+105);
UKS_line(u1-58,v1+108,u2-17,v2+108);
UKS_line(u1-56,v1+111,u2-19,v2+111);
UKS_line(u1-50,v1+20,u2,v2);
UKS_line(u1-25,v1,u2-3,v2+8);
UKS_line(u1-25,v1,u2-9,v2+2);
UKS_outtextxy(u1-53,v1+69,"+");
UKS_outtextxy(u1-53,v1+75,"-");
UKS_circle(u1-50,v1+20,3);
UKS_circle(u1-50,v1+20,2);
UKS_circle(u1-50,v1+20,1);
UKS_circle(u1-25,v1,3);
UKS_circle(u1-25,v1,2);
UKS_circle(u1-25,v1,1);
UKS_circle(u1-75,v1,3);
UKS_circle(u1-75,v1,2);
UKS_circle(u1-75,v1,1);
UKS_outtextxy(u1-40,v1-10,"e");
UKS_outtextxy(u1-70,v1-10,"o");
}
void iletkenlik(float u1,float v1)
{
float u2,v2;
v2=v1;
u2=u1-25;
UKS_line(u1,v1,u2,v2);
UKS_line(u1-75,v1,u2-75,v2);
UKS_line(u1-100,v1,u2-75,v2+10);
UKS_line(u1-110,v1+10,u2-65,v2+10);

```

```

UKS_line(u1-108,v1+13,u2-67,v2+13);
UKS_line(u1-106,v1+16,u2-69,v2+16);
UKS_line(u1-50,v1+20,u2-25,v2+40);
UKS_line(u1-60,v1+40,u2-15,v2+40);
UKS_line(u1-60,v1+45,u2-15,v2+45);
UKS_line(u1-50,v1+45,u2-25,v2+65);
UKS_line(u1-60,v1+65,u2-15,v2+65);
UKS_line(u1-58,v1+68,u2-17,v2+68);
UKS_line(u1-56,v1+71,u2-19,v2+71);
UKS_line(u1-50,v1+20,u2,v2);
UKS_line(u1-25,v1,u2-3,v2+8);
UKS_line(u1-25,v1,u2-9,v2+2);
UKS_circle(u1-50,v1+20,3);
UKS_circle(u1-50,v1+20,2);
UKS_circle(u1-50,v1+20,1);
UKS_circle(u1-25,v1,3);
UKS_circle(u1-25,v1,2);
UKS_circle(u1-25,v1,1);
UKS_circle(u1-75,v1,3);
UKS_circle(u1-75,v1,2);
UKS_circle(u1-75,v1,1);
UKS_outtextxy(u1-40,v1-10,"e");
UKS_outtextxy(u1-70,v1-10,"o");
}
void kapasite(float u1,float v1)
{
    float u2,v2;
    u2=u1;
    v2=v1+50;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-10,v1+50,u2+10,v2);
    UKS_line(u1-10,v1+55,u2+10,v2+5);
    UKS_line(u1,v1+55,u2,v2+55);
    UKS_line(u1-10,v1+105,u2+10,v2+55);
    UKS_line(u1-8,v1+108,u2+8,v2+58);
    UKS_line(u1-6,v1+111,u2+6,v2+61);
}
void iletkenlik1 (float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1+30;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1+80,v1,u2+65,v2);
    UKS_line(u1+95,v1,u2+65,v2+20);
    UKS_line(u1+85,v1+20,u2+75,v2+20);
    UKS_line(u1+87,v1+23,u2+73,v2+23);
    UKS_line(u1+89,v1+26,u2+71,v2+26);
    UKS_line(u1+55,v1+20,u2+25,v2+40);
    UKS_line(u1+45,v1+40,u2+35,v2+40);
    UKS_line(u1+45,v1+45,u2+35,v2+45);
    UKS_line(u1+55,v1+45,u2+25,v2+65);
    UKS_line(u1+45,v1+65,u2+35,v2+65);
    UKS_line(u1+47,v1+68,u2+33,v2+68);
    UKS_line(u1+49,v1+71,u2+31,v2+71);
    UKS_line(u1+55,v1+20,u2,v2);
    UKS_line(u1+30,v1,u2+3,v2+8);
    UKS_line(u1+30,v1,u2+9,v2+2);

```

```

    UKS_circle(u1+30,v1,3);
    UKS_circle(u1+30,v1,2);
    UKS_circle(u1+30,v1,1);
    UKS_circle(u1+80,v1,3);
    UKS_circle(u1+80,v1,2);
    UKS_circle(u1+80,v1,1);
    UKS_circle(u1+55,v1+20,3);
    UKS_circle(u1+55,v1+20,2);
    UKS_circle(u1+55,v1+20,1);
    UKS_outtextxy(u1+35,v1-10,"e");
    UKS_outtextxy(u1+67,v1-10,"o");
}
void dugum(float u1,float v1)
{
    UKS_circle(u1,v1,3);
    UKS_circle(u1,v1,2);
    UKS_circle(u1,v1,1);
}
void main()
{
    int GrAdapter,GrMod,bitti=0;
    int hata_kodu;
    detectgraph ( &GrAdapter,&GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter , &GrMod ,"c:\\borlandc\\bgi" );
    hata_kodu = graphresult();
    if(hata_kodu != grOk)
    {
        printf("grafik hatası : (#%d kodlu hata)! %s\n",hata_kodu,grapherrormsg(hata_kodu));
        getch();
        exit(1);
    }
    setbkcolor( 1 );
    UKS_rectangle(200,300,250,350);
    UKS_rectangle(300,100,350,150);
    UKS_rectangle(400,300,450,350);

    UKS_outtextxy(200,280,"CCII-2");
    UKS_outtextxy(300,80,"CCII-1");
    UKS_outtextxy(400,280,"CCII-3");

    UKS_line(250,310,400,310);
    UKS_line(285,310,285,325);
    UKS_line(350,125,485,125);
    UKS_line(485,125,485,325);
    UKS_outtextxy(190,105,"VIN");
    dugum(225,110);
    UKS_line(600,325,620,325);
    dugum(620,325);
    UKS_outtextxy(630,320,"VOUT");
    UKS_outtextxy(205,310,"Y");
    UKS_outtextxy(205,335,"X");
    UKS_outtextxy(240,310,"Z");
    UKS_outtextxy(222,325,"+");
    UKS_outtextxy(305,110,"Y");
    UKS_outtextxy(305,135,"X");
    UKS_outtextxy(340,125,"Z");
    UKS_outtextxy(322,125,"+");
    UKS_outtextxy(405,310,"Y");

```

```

UKS_outtextxy(405,335,"X");
UKS_outtextxy(440,325,"Z");
UKS_outtextxy(422,325,"-");
UKS_outtextxy(347,409,"+");
UKS_outtextxy(347,415,"-");
UKS_outtextxy(370,412,"Vx3");
UKS_outtextxy(300,375,"Ci3");
UKS_outtextxy(290,180,"Ci1");
UKS_outtextxy(500,375,"Ci5");
UKS_line(450,325,600,325);
UKS_line(600,325,600,460);
UKS_line(600,460,50,460);
UKS_line(225,110,300,110);
UKS_line(275,140,300,140);
UKS_line(50,460,50,310);
UKS_line(50,310,200,310);

UKS_outtextxy(555,380,"Ci6");
UKS_outtextxy(165,380,"Ci2");
UKS_outtextxy(365,380,"Ci4");

iletkenlik(200,340);
kapasite(275,140);
kapasite(285,325);
kapasite(485,325);
gecikmeli_iletkenlik(400,340);
iletkenlik1(485,340);
dugum(285,310);
dugum(485,325);
/*****/
UKS_rectangle(200,550,250,600);
UKS_line(200,560,150,560);
UKS_line(200,590,150,590);
UKS_line(250,575,385,575);
dugum(385,575);
UKS_outtextxy(395,570,"VOUT");
dugum(150,560);
UKS_outtextxy(115,555,"VIN");
UKS_outtextxy(205,560,"Y");
UKS_outtextxy(205,585,"X");
UKS_outtextxy(240,575,"Z");
UKS_outtextxy(222,575,"+");
kapasite(150,591);
UKS_outtextxy(169,640,"C01");
kapasite(375,586);
UKS_line(375,576,375,586);
UKS_outtextxy(285,640,"C02");
gecikmeli_iletkenlik(375,601);
UKS_outtextxy(390,630,"C03");
iletkenlik1( 375,601);
UKS_outtextxy(445,640,"C02");
UKS_outtextxy(290,670,"Vz");
UKS_outtextxy(560,120,"2.DERECE CCI-SC YÜKSEK");
UKS_outtextxy(570,140,"GEÇİREN AKTİF FİLTRE");
UKS_outtextxy(610,160,"BİRİM YAPISI");
UKS_outtextxy(610,180,"i=1,2,3,...N");
UKS_outtextxy(560,580,"1.DERECE CCI-SC YÜKSEK");
UKS_outtextxy(570,600,"GEÇİREN AKTİF FİLTRE");
UKS_outtextxy(610,620,"BİRİM YAPISI");

```

```

do{
switch(tusal())
{
case ESC :
        bitti=1;
        break;

default :
        printf("\a");
        break;

}
}while(!bitti);
closegraph();
}

```

DEVREB.CPP

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<bios.h>
#include<graphics.h>
#define ESC 27
int tusal(void)
{
        int TUS,lo, hi;
        TUS=bioskey(0);
        lo=TUS & 0X00FF;
        hi=(TUS & 0XFF00) >> 8;
        return((lo==0) ? hi+256 : lo);
}
int u_x(float u)
{
        return((int)(0.79875*u));
}
int v_y(float v)
{
        return((int)(0.59875*v));
}
void UKS_line (float u1, float v1, float u2,float v2)
{
        line ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_circle (float u, float v, float r)
{
        circle ( u_x(u), v_y(v), u_x(r) );
}
void UKS_rectangle (float u1, float v1, float u2,float v2)
{
        rectangle ( u_x(u1), v_y(v1), u_x(u2),v_y(v2));
}
void UKS_outtextxy(float u, float v,char *s)
{
        outtextxy(u_x(u),v_y(v), s);
}
void gecikmeli_iletkenlik(float u1,float v1)
{
        float u2,v2;
        v2=v1;
        u2=u1-25;
}

```

```

UKS_line(u1,v1,u2,v2);
UKS_line(u1-75,v1,u2-75,v2);
UKS_line(u1-100,v1,u2-75,v2+10);
UKS_line(u1-110,v1+10,u2-65,v2+10);
UKS_line(u1-108,v1+13,u2-67,v2+13);
UKS_line(u1-106,v1+16,u2-69,v2+16);
UKS_line(u1-50,v1+20,u2-25,v2+40);
UKS_line(u1-60,v1+40,u2-15,v2+40);
UKS_line(u1-60,v1+45,u2-15,v2+45);
UKS_line(u1-50,v1+45,u2-25,v1+65);
UKS_line(u1-50,v1+65,u2-10,v2+75);
UKS_line(u1-50,v1+65,u2-40,v2+75);
UKS_line(u1-65,v1+75,u2-25,v2+85);
UKS_line(u1-50,v1+85,u2-10,v2+75);
UKS_line(u1-50,v1+85,u2-25,v2+105);
UKS_line(u1-60,v1+105,u2-15,v2+105);
UKS_line(u1-58,v1+108,u2-17,v2+108);
UKS_line(u1-56,v1+111,u2-19,v2+111);
UKS_line(u1-50,v1+20,u2,v2);
UKS_line(u1-25,v1,u2-3,v2+8);
UKS_line(u1-25,v1,u2-9,v2+2);
UKS_outtextxy(u1-53,v1+69,"+");
UKS_outtextxy(u1-53,v1+75,"-");
UKS_circle(u1-50,v1+20,3);
UKS_circle(u1-50,v1+20,2);
UKS_circle(u1-50,v1+20,1);
UKS_circle(u1-25,v1,3);
UKS_circle(u1-25,v1,2);
UKS_circle(u1-25,v1,1);
UKS_circle(u1-75,v1,3);
UKS_circle(u1-75,v1,2);
UKS_circle(u1-75,v1,1);
UKS_outtextxy(u1-40,v1-10,"e");
UKS_outtextxy(u1-70,v1-10,"o");
}
void iletkenlik(float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1-25;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-75,v1,u2-75,v2);
    UKS_line(u1-100,v1,u2-75,v2+10);
    UKS_line(u1-110,v1+10,u2-65,v2+10);
    UKS_line(u1-108,v1+13,u2-67,v2+13);
    UKS_line(u1-106,v1+16,u2-69,v2+16);
    UKS_line(u1-50,v1+20,u2-25,v2+40);
    UKS_line(u1-60,v1+40,u2-15,v2+40);
    UKS_line(u1-60,v1+45,u2-15,v2+45);
    UKS_line(u1-50,v1+45,u2-25,v2+65);
    UKS_line(u1-60,v1+65,u2-15,v2+65);
    UKS_line(u1-58,v1+68,u2-17,v2+68);
    UKS_line(u1-56,v1+71,u2-19,v2+71);
    UKS_line(u1-50,v1+20,u2,v2);
    UKS_line(u1-25,v1,u2-3,v2+8);
    UKS_line(u1-25,v1,u2-9,v2+2);
    UKS_circle(u1-50,v1+20,3);
    UKS_circle(u1-50,v1+20,2);
    UKS_circle(u1-50,v1+20,1);

```

```

    UKS_circle(u1-25,v1,3);
    UKS_circle(u1-25,v1,2);
    UKS_circle(u1-25,v1,1);
    UKS_circle(u1-75,v1,3);
    UKS_circle(u1-75,v1,2);
    UKS_circle(u1-75,v1,1);
    UKS_outtextxy(u1-40,v1-10,"e");
    UKS_outtextxy(u1-70,v1-10,"o");
}
void kapasite(float u1,float v1)
{
    float u2,v2;
    u2=u1;
    v2=v1+50;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1-10,v1+50,u2+10,v2);
    UKS_line(u1-10,v1+55,u2+10,v2+5);
    UKS_line(u1,v1+55,u2,v2+55);
    UKS_line(u1-10,v1+105,u2+10,v2+55);
    UKS_line(u1-8,v1+108,u2+8,v2+58);
    UKS_line(u1-6,v1+111,u2+6,v2+61);
}
void iletkenlik1 (float u1,float v1)
{
    float u2,v2;
    v2=v1;
    u2=u1+30;
    UKS_line(u1,v1,u2,v2);
    UKS_line(u1+80,v1,u2+65,v2);
    UKS_line(u1+95,v1,u2+65,v2+20);
    UKS_line(u1+85,v1+20,u2+75,v2+20);
    UKS_line(u1+87,v1+23,u2+73,v2+23);
    UKS_line(u1+89,v1+26,u2+71,v2+26);
    UKS_line(u1+55,v1+20,u2+25,v2+40);
    UKS_line(u1+45,v1+40,u2+35,v2+40);
    UKS_line(u1+45,v1+45,u2+35,v2+45);
    UKS_line(u1+55,v1+45,u2+25,v2+65);
    UKS_line(u1+45,v1+65,u2+35,v2+65);
    UKS_line(u1+47,v1+68,u2+33,v2+68);
    UKS_line(u1+49,v1+71,u2+31,v2+71);
    UKS_line(u1+55,v1+20,u2,v2);
    UKS_line(u1+30,v1,u2+3,v2+8);
    UKS_line(u1+30,v1,u2+9,v2+2);

    UKS_circle(u1+30,v1,3);
    UKS_circle(u1+30,v1,2);
    UKS_circle(u1+30,v1,1);
    UKS_circle(u1+80,v1,3);
    UKS_circle(u1+80,v1,2);
    UKS_circle(u1+80,v1,1);
    UKS_circle(u1+55,v1+20,3);
    UKS_circle(u1+55,v1+20,2);
    UKS_circle(u1+55,v1+20,1);
    UKS_outtextxy(u1+35,v1-10,"e");
    UKS_outtextxy(u1+67,v1-10,"o");
}
void dugum(float u1,float v1)
{
    UKS_circle(u1,v1,3);

```

```

    UKS_circle(u1,v1,2);
    UKS_circle(u1,v1,1);
}
void main()
{
    int GrAdapter,GrMod,bitti=0;
    int hata_kodu;
    detectgraph ( &GrAdapter,&GrMod );
    GrAdapter = DETECT;
    initgraph( &GrAdapter , &GrMod ,"c:\\borlandc\\bgi" );
    hata_kodu = graphresult();
    if(hata_kodu != grOk)
    {
        printf("grafik hatası : (#%d kodlu hata)! %s\n",hata_kodu,grapherrormsg(hata_kodu));
        getch();
        exit(1);
    }
    setbkcolor( 1 );

    UKS_rectangle(200,300,250,350);
    UKS_rectangle(300,100,350,150);
    UKS_rectangle(400,300,450,350);

    UKS_outtextxy(200,280,"CCII-2");
    UKS_outtextxy(300,80,"CCII-1");
    UKS_outtextxy(400,280,"CCII-3");

    UKS_line(250,310,400,310);
    UKS_line(285,310,285,325);
    UKS_line(350,125,485,125);
    UKS_line(485,125,485,325);
    UKS_outtextxy(190,105,"VIN");
    dugum(225,110);
    UKS_line(600,325,620,325);
    UKS_outtextxy(630,320,"VOUT");
    dugum(620,325);
    UKS_outtextxy(205,310,"Y");
    UKS_outtextxy(205,335,"X");
    UKS_outtextxy(240,310,"Z");
    UKS_outtextxy(222,325,"+");
    UKS_outtextxy(305,110,"Y");
    UKS_outtextxy(305,135,"X");
    UKS_outtextxy(340,125,"Z");
    UKS_outtextxy(322,125,"+");
    UKS_outtextxy(405,310,"Y");
    UKS_outtextxy(405,335,"X");
    UKS_outtextxy(440,325,"Z");
    UKS_outtextxy(422,325,"-");
    UKS_outtextxy(347,409,"+");
    UKS_outtextxy(347,415,"-");
    UKS_outtextxy(370,412,"Vx3");
    UKS_outtextxy(300,375,"Ci4");
    UKS_outtextxy(500,375,"Ci6");
    UKS_outtextxy(120,230,"Vx1");
    UKS_outtextxy(120,200,"Ci1");
    UKS_outtextxy(250,200,"Ci2");
    UKS_line(450,325,600,325);
    UKS_line(600,325,600,460);
    UKS_line(600,460,50,460);

```

```

UKS_line(225,110,300,110);
UKS_line(150,140,300,140);
UKS_line(150,140,150,160);
UKS_line(280,140,280,160);
UKS_line(50,460,50,310);
UKS_line(50,310,200,310);

UKS_outtextxy(555,380,"Ci7");
UKS_outtextxy(165,380,"Ci3");
UKS_outtextxy(365,380,"Ci5");

iletkenlik(200,340);
gecikmeli_iletkenlik(150,160);
iletkenlik(280,160);
kapasite(285,325);
kapasite(485,325);
gecikmeli_iletkenlik(400,340);
iletkenlik1(485,340);
dugum(285,310);
dugum(485,325);
UKS_outtextxy(560,120,"2.DERECE CCII-SC BAND");
UKS_outtextxy(570,140,"GEÇİREN AKTİF FİLTRE");
UKS_outtextxy(610,160,"BİRİM YAPISI");
UKS_outtextxy(610,180,"i=1,2,3,...N");
do{
switch(tusal())
{
case ESC :
        bitti=1;
        break;

default :
        printf("\a");
        break;

}
}while(!bitti);
closegraph();
}

```

CURSOR.H

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<graphics.h>
#define TRUE 1
#define FALSE 0
#define ESC_KEY 27
#define ENTER_KEY 13

void ilkcursor (int x, int y,int boyut)
{
    line(x-2*boyut, y, x+2*boyut, y);
    line(x, y-2*boyut, x, y+2*boyut);
}

int grcursor(float fxx[], float fyy[],int x, int y, int boyut, int adim,int eky[], int ekx[],
float max, char cikistusu, int *sonx, int *sony)
{
    char ktr[31],ktr1[31];
    int xx,yy,i=0,j;

```

```

int k1,k2;
struct viewporttype winbilgi;
struct linesettingstype ilkcizimstili;
int scrx,scry;
getviewsettings(&winbilgi);
getlinesettings(&ilkcizimstili);
scrx=winbilgi.right-winbilgi.left;
scry=winbilgi.bottom-winbilgi.top;
setlinestyle(1,10,1);
setwritemode(XOR_PUT);

xx=x;
yy=y;

do{
    ilkcursor(xx,yy,boyut);
    k1=0;
    k2=0;
    k1=getch();
    if(k1==0) k2=getch();
    switch(k1)
    {
        case '+':
            adim++;
            if(adim>20) adim=20;
            break;
        case '-':
            adim--;
            if(adim<1) adim=1;
            break;
        case '1': case '2':
        case '3': case '4':
        case '5': case '6':
        case '7': case '8':
        case '9':
            adim=k1-48;
    }
    switch(k2)
    {
        case 77 :
            if(i<198){
                i++;
                x=ekx[i];
                y=eky[i];
                setviewport(450,440,639,479,0);
                clearviewport();
                outtextxy(1,1," f=");
                outtextxy(1,20," | H |=");
                sprintf(ktr,"%0.3f Hz",fx[i]);
                sprintf(ktr1,"%0.3f",fyy[i]);
                outtextxy(35,1,ktr);
                outtextxy(35,20,ktr1);
                setviewport(67,10,637,400,1);
            }
            if(x>max) x=max;
            if(y>scry) y=scry;
            break;
        case 75 :
            if(i>0){

```

```

        i--;
        x=ekx[i];
        y=eky[i];
        setviewport(450,440,639,479,0);
        clearviewport();
        outtextxy(1,1, " f=");
        outtextxy(1,20," | H |=");
        sprintf(ktr,"%0.3f Hz",fxx[i]);
        sprintf(ktr1,"%0.3f",fyy[i]);
        outtextxy(35,1,ktr);
        outtextxy(35,20,ktr1);
        setviewport(67,10,637,400,1);
    }
    if(x<0) x=0;
    break;
}
ilkcursor(xx,yy,boyut);
xx=x;
yy=y;
}while((k1 != cikistusu) && (k1 != ESC_KEY));
setlinestyle(ilkcizimstili.linestyle, ilkcizimstili.upattern, ilkcizimstili.thickness);
setwritemode(COPY_PUT);
*sonx=x;
*sony=y;
if(k1==ESC_KEY) return TRUE;
else return FALSE;
}

```

CURSOR1.H

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<graphics.h>
#define TRUE 1
#define FALSE 0
#define ESC_KEY 27
#define ENTER_KEY 13

void ilkcursor (int x, int y,int boyut)
{
    line(x-2*boyut, y, x+2*boyut, y);
    line(x, y-2*boyut, x, y+2*boyut);
}

int grcursor(float fxx[],float fyy[],int x, int y, int boyut, int adim,int eky[], int ekx[],
float max, char cikistusu, int *sonx, int *sony)
{
    char ktr[31],ktr1[31];
    int xx,yy,i=0,j;
    int k1,k2;
    struct viewporttype winbilgi;
    struct linesettingstype ilkcizimstili;
    int scrx,scry;
    getviewsettings(&winbilgi);
    getlinesettings(&ilkcizimstili);
    scrx=winbilgi.right-winbilgi.left;
    scry=winbilgi.bottom-winbilgi.top;
    setlinestyle(DOTTED_LINE, 10 , NORM_WIDTH);
    setwritemode(XOR_PUT);
}

```

```

xx=x;
yy=y;

do{
    ilkcursor(xx,yy,boyut);
    k1=0;
    k2=0;
    k1=getch();
    if(k1==0) k2=getch();
    switch(k1)
    {
        case '+':
            adim++;
            if(adim>20) adim=20;
            break;

        case '-':
            adim--;
            if(adim<1) adim=1;
            break;

        case '1': case '2':
        case '3': case '4':
        case '5': case '6':
        case '7': case '8':
        case '9':
            adim=k1-48;

    }
    switch(k2)
    {
        case 77 :
            if(i<399){
                i++;
                x=ekx[i];
                y=eky[i];
                setviewport(450,440,639,479,0);
                clearviewport();
                outtextxy(1,1, " f=");
                outtextxy(1,20," | H |=");
                sprintf(ktr,"%0.3f Hz",fxx[i]);
                sprintf(ktr1,"%0.3f",fyy[i]);
                outtextxy(35,1,ktr);
                outtextxy(35,20,ktr1);
                setviewport(67,10,637,400,1);
            }
            if(x>scrx) x=scrx;
            if(y>scry) y=scry;
            break;

        case 75 :
            if(i>0){
                i--;
                x=ekx[i];
                y=eky[i];
                setviewport(450,440,639,479,0);
                clearviewport();
                outtextxy(1,1, " f=");
                outtextxy(1,20," | H |=");
                sprintf(ktr,"%0.3f Hz",fxx[i]);
                sprintf(ktr1,"%0.3f",fyy[i]);
                outtextxy(35,1,ktr);
            }
    }
}

```

```
        outtextxy(35,20,ktr1);
        setviewport(67,10,637,400,1);
    }
    if(x<0) x=0;
    break;
}
ilkcursor(xx,yy,boyut);
xx=x;
yy=y;
}while((k1 != cikistusu) && (k1 != ESC_KEY));
setlinestyle(ilkcizimstili.linestyle, ilkcizimstili.upattern, ilkcizimstili.thickness);
setwritemode(COPY_PUT);
*sonx=x;
*sony=y;
if(k1==ESC_KEY) return TRUE;
else return FALSE;
}
```



ÖZGEÇMİŞ

Hakan Gürkan, 1973 yılında İstanbul'da doğdu. Lise eğitimini Kenan Evren Lisesinde yaptı ve bu liseyi ikincilikle bitirdi. 1990 yılında İstanbul Teknik Üniversitesi Elektrik-Elektronik Fakültesi Elektronik ve Haberleşme Mühendisliği bölümünde lisans eğitimine başladı. 1994 yılında bu bölümden mezun oldu ve aynı yıl İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Ana Bilim Dalı Elektronik ve Haberleşme Mühendisliği programında yüksek lisans eğitimine başladı.

