SAYISAL MANTIK DEVRELERİNİN BİLGİSAYARDA BENZETİMİ

HASAN TEMURTAŞ

Dumlupınar Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Elektrik Elektronik Mühendisliği Anabilim Dalında
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

Danışman :Yrd. Doç. Dr. M.Ali EBEOĞLU

Mart-1996

Hasan TEMURTAŞ'ın YÜKSEK LİSANS tezi olarak hazırladığı "Sayısal Mantık Devrelerinin Bilgisayarda Benzetimi" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

27./05./1996

Üye : Prof.Dr. Atila Barkana

Üye : Prof.Dr. Hamdi Atmaca

Üye : Yrd.Doç.Dr. Mehmet Ali EBEOĞLU

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 13.06.1996 gün ve ....08..... sayılı kararıyla onaylanmıştır.

Fen Bilimleri Enstitüsü Müdürü
Doç.Dr. İ. Göktay EDİZ

ÖZET

Yüksek Lisans Tezi

SAYISAL MANTIK DEVRELERİNİN BİLGİSAYARDA BENZETİMİ

Bu tezin konusu, sayısal mantık devrelerinin bilgisayarda modellenmesi ve bütün mümkün şartlar altındaki davranışlarının incelenmesi ile ilgili PASCAL Programlama Dilinde genel amaçlı bir yazılımın tasarlanıp gerçekleştirilmesidir.

Bu benzetim programında AND, NAND, OR, NOR, EXOR, EXNOR ve NOT Kapıları, JK, D, T ve RS Flip-Flopları, Yarım Toplayıcı (H.A), Tam Toplayıcı (F.A), 1*2'lik Kod Çözücü (Decoder), 2*4'lük Kod Çözücü, 2*1'lik Kodlayıcı (Encoder), 4*2'lik Kodlayıcı, 2*1'lik Bilgi Seçici (MUX), 4*1'lik Bilgi Seçici, 1*2'lik Bilgi Dağıtıcı (DEMUX), 1*4'lük Bilgi Dağıtıcı, 2 bit'lik Kaydedici (Register), 4 bit'lik Kaydedici gibi devre elemanlarının benzetimi kullanılabilmektedir.

Herhangi bir devre elemanının yerleştirilmesi, iptal edilmesi veya yerinin değiştirilmesi, devre elemanları arasına hatların döşenmesi veya iptal edilmesi, devre elemanlarının girişlerine girdilerin yerleştirilmesi veya iptal edilmesi, herhangi bir devre elemanının herhangi bir çıkışı üzerindeki çıktının hesaplanması, ekrandaki devrenin sağa, sola, yukarı, aşağı kaydırılması, çizilen bir devrenin kaydedilmesi veya kaydedilmiş herhangi bir devrenin geri çağırılması gibi işlemler fare (mouse) ve klavye (keyboard) yardımıyla kolayca yapılmaktadır.

Programın gerçekleştirilmesinde 30*20'lik bir A matrisi kullanılmıştır. PUT Altprogramı bu matristeki sayısal değerleri kullanarak devrenin şeklini çizer. Devrenin her değişmesi matrisin değişmesi demektir. Bu yüzden matrisin her değişmesinde PUT Altprogramı tekrar çalışarak önceki devre şeklini silip yeni devre şeklini çizer.

Anahtar Kelimeler: Benzetim, Sayısal Mantık Devreleri, Yazılım, Tasarım.

SUMMARY

M. Sc. Thesis

COMPUTER SIMULATION OF DIGITAL LOGIC CIRCUITS

The main purpose of this thesis is to design and implement a general-purpose software which is related in modelling of digital logic circuits on computer screen and examine the behavior of them at any possible conditions by using PASCAL programming language.

In this simulation program, simulation of the circuit elements such as AND, NAND, OR, NOR, EXOR, EXNOR and NOT gates, JK, D, T and RS Flip-Flops, Half Adder, Full Adder, 1*2 Decoder, 2*4 Decoder, 2*1 Encoder, 4*2 Encoder, 2*1 Multiplexer, 4*1 Multiplexer, 1*2 Demultiplexer, 1*4 Demultiplexer, 2 bit Register, 4 bit Register can be used.

To put or cancel any circuit element, to move any circuit element from one place to another, to put or cancel the wires among the circuit elements, to put or cancel inputs to the inputs of the circuit elements, to find the output on any output of any citcuit element, to move to the circuit in the screen to the right, left, up or down to register any circuit was drawn or to recall any circuit was registered are easily done by means of a mouse or a keyboard.

Out in the implementation of the program, an A (20*20) matrix is used. The subroutine PUT draws the shape of the circuit by using numerical values from this matrix. Every change of the circuit means a change of the matrix. Therefore, by execution of the subroutine PUT, for any change of the matrix, the subroutine PUT is reexecuted to delete the preceding circuit and draw the new shape of the circuit.

Keywords: Simulation, Digital Logic Circuits, Software, Design.

TEŞEKKÜR

Bu çalışma Dumlupınar Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölümünden Sayın Yrd. Doç. Dr. Mehmet Ali EBEOĞLU'nun yöneticiliğinde yapılmıştır. Çok yakın ilgi ve yardımlarından dolayı kendisine teşekkür ederim.

Ayrıca Dumlupınar Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölüm Başkanı Sayın Prof. Dr. Hamdi ATMACA'ya ve Anadolu Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölümü Öğretim Üyesi Sayın Prof. Dr. Atila BARKANA'ya göstermiş oldukları ilgi ve yardımlarından dolayı teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET III

SUMMARY IV

TEŞEKKÜR V

ŞEKİLLER VIII

TABLOLAR IX

1. GİRİŞ 1

2. SAYISAL MANTIK DEVRELERİ 3

2.1. Kapılar 3
2.1.1. AND kapısı 3
2.1.2. OR kapısı 4
2.1.3. NOT kapısı 4
2.1.4. NAND kapısı 4
2.1.5. NOR kapısı 5
2.1.6. EXOR kapısı 5
2.1.7. EXNOR kapısı 5
2.2. Flip-Floplar 6
2.2.1. RS Flip-Flop 7
2.2.2. D Flip-Flop 8
2.2.3. JK Flip-Flop 8
2.2.4. T Flip-Flop 9
2.3. Toplayıcılar 9
2.4. Kod Çözücüler 10
2.5. Kodlayıcılar 10
2.6. Bilgi Seçiciler 10
2.7. Bilgi Dağıtıcılar 11
2.8. Kaydediciler 11

3. PASCAL PROGRAMLAMA DİLİ 12

3.1. Veri Tipleri 12

İÇİNDEKİLER(devam)

Sayfa

3.2.1. Etiket tanımlama bloğu 13
3.2.2. Sabit tanımlama bloğu 14
3.2.3. Değişken tanımlama bloğu 14
3.2.4. Tip tanımlama bloğu 14
3.2.5. Fonksiyon tanımlama bloğu 15
3.2.6. Prosedür tanımlama bloğu 15
3.3. Aritmetik Operatörler 15
3.4. Atama İşlemi 16
3.5. Arşiv Fonksiyonları 16
3.6. Karakter İşlemleri 16
3.7. Akış Deyimleri Ve Döngüler 17
3.7.1. GOTO deyimi 17
3.7.2. IF deyimi 17
3.7.3. CASE OF deyimi 18
3.7.4. FOR döngüsü 18
3.7.5. REPEAT döngüsü 18
3.7.6. WHILE döngüsü 19
3.8. Diziler 19
3.9. Dosyalama 19
3.9.1. Sıralı erişimli dosyalar 20
3.10. Grafik Ekran Ve Mouse 22

4. SAYISAL MANTIK DEVRELERİ İLE İLGİLİ BENZETİM PROGRAMI 25

4.1. Programda Kullanılan Önemli Altprogramlar 28
4.2. Örnek Program 30

5. SONUÇ VE TARTIŞMA 37

KAYNAKLAR 38

EKLER

1. ALTPROGRAMLARIN AKIŞ DİYAGRAMLARI

2. PROGRAMIN LİSTESİ

ŞEKİLLER

Şekil	sayfa
2.1. İki girişli bir AND kapısı ve doğruluk tablosu	4
2.2. İki girişli bir OR kapısı ve doğruluk tablosu	4
2.3. Bir NOT kapısı ve doğruluk tablosu	4
2.4. İki girişli bir NAND kapısı ve doğruluk tablosu	5
2.5. İki girişli bir NOR kapısı ve doğruluk tablosu	5
2.6. İki girişli bir EXOR kapısı, açık gösterimi ve doğruluk tablosu	5
2.7. İki girişli bir EXNOR kapısı, açık gösterimi ve doğruluk tablosu	6
2.8. NOR kapılarıyla kurulan temel Flip-Flop devre ve doğruluk tablosu	6
2.9. NAND kapılarıyla kurulan temel Flip-Flop devre ve doğruluk tablosu	7
2.10. Clock-Pulse jenaratörü tarafından üretilen CP sinyali	7
2.11. Bir RS Flip-Flop, açık gösterimi ve doğruluk tablosu	8
2.12. Bir D Flip-Flop ve doğruluk tablosu	8
2.13. Bir JK Flip-Flop, açık gösterimi ve doğruluk tablosu	9
2.14. Bir T Flip-Flop ve doğruluk tablosu	9
2.15. Bir Yarım Toplayıcı ve bir Tam Toplayıcı	10
2.16. 2*4'lük bir Kod Çözücü ve doğruluk tablosu	10
2.17. 4*2'lik bir Kodlayıcı ve doğruluk tablosu	10
2.18. 4*1'lik bir Bilgi Seçici ve doğruluk tablosu	11
2.19. 1*4'lük bir Bilgi Dağıtıcı ve doğruluk tablosu	11
2.20. 4 bit'lik bir Kaydedici	11
4.1. Bir Sayısal Mantık Devresinin Blok Diyagramı	31
4.2. Herhangi bir devre elemanının çıkışının 30*200'lük B matrisinde gösterimi için yararlanılacak kurallar	33

TABLOLAR

Tablo		sayfa
4.1.	T'deki değişme	26
4.2.	Şekil 4.1'deki devrenin 30*20'lik A matrisindeki gösterimi.	31
4.3.	Tablo 4.2'deki 30*20'lik A matrisinin yardımıyla elde edilen 30*20'lik O matrisi	33

1. GİRİŞ

En yaygın bilgisayar uygulamalarından biri de benzetimdir (simulation). Benzetim programlarının genel amacı, gerçekte veya hayal gücünde var olan olguların benzetiminin bilgisayar yardımıyla gerçekleştirilmesidir. Teknolojinin hızlı bir şekilde ilerlemesiyle birlikte tasarım problemlerinin büyüklüğü ve karmaşıklığı da gittikçe artmaktadır. Bu sebeble problemleri hafifletmek için bu amaçla yapılmış benzetim programlarına ihtiyaç vardır. Sayısal mantık devreleri ile ilgili yapılan benzetim programlarının genel amacı, sayısal mantık devrelerinin bilgisayar ekranında modellenmesi ve bütün mümkün şartlar altındaki davranışlarının incelenmesidir. Bu konuyla ilgili Orta Doğu Teknik Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölümünde 1983 tarihinde Uğur HALICI ve 1987 tarihinde Atamazhori PARASİMA tarafından yapılmış yüksek lisans tez çalışmaları vardır.

Diğer devrelerde olduğu gibi sayısal mantık devrelerinin de labaratuvar ortamında kurulup gerçekleştirilmesi ve sonuçlarının gözlemlenmesi zordur; çünkü hem zaman alır ve hem de devre öğelerini bulmak kolay değildir. Bu çalışmada sayısal mantık devreleri ile ilgili bir benzetim programı Pascal Dili kullanılarak yazılmıştır.

Programda AND(VE), NAND(VE DEĞİL), OR(VEYA), NOR(VEYA DEĞİL), EXOR(AYRICALIKLI VEYA), EXNOR(EXOR'UN DEĞİLİ) ve NOT(DEĞİL) kapıları, JK, D, T ve RS Flip-Flopları, Yarım Toplayıcı(Half Adder), Tam Toplayıcı(Full Adder), 1*2'lik Kod Çözücü(1*2 Decoder), 2*4'lük Kod Çözücü(2*4 Decoder), 2*1'lik Kodlayıcı(2*1 Encoder), 4*2'lik Kodlayıcı(4*2 Encoder), 2*1'lik Bilgi Seçici(2*1 Multiplexer), 4*1'lik Bilgi Seçici(4*1 Multiplexer), 1*2'lik Bilgi Dağıtıcı(1*2 Demultiplexer), 1*4'lük Bilgi Dağıtıcı(1*4 Demultiplexer), 2 bit'lik Kaydedici(2 bit Register) ve 4 bit'lik Kaydedici(4 bit Register) devre elemanlarının benzetimi kullanılmaktadır. Bu program yardımıyla yukardaki devre elemanlarının bileşiminden meydana gelen her devrenin şekli rahatlıkla oluşturulur ve istenilen herhangi bir devre elemanının herhangi bir çıktısı rahatlıkla hesaplanır. Çıktı hesaplanırken girdilerde eksiklik varsa, program uyarı vermektedir.

Herhangi bir devre elemanının yerleştirilmesi, iptal edilmesi veya yerinin değiştirilmesi, devre elemanları arasına hatların döşenmesi veya iptal edilmesi, devre elemanlarının girişlerine girdilerin yerleştirilmesi veya iptal edilmesi, herhangi bir devre elemanının herhangi bir çıkışı üzerindeki çıktının hesaplanması, ekrandaki devrenin sağa, sola, yukarı, aşağı kaydırılması, çizilen bir devrenin kaydedilmesi veya kaydedilmiş herhangi bir devrenin geri çağırılması gibi işlemler fare (mouse) ve klavye (keyboard) yardımıyla kolayca yapılabilmektedir.

Bu programın temel mantığı matrisler üzerine kurulmuştur. Devre hakkında bütün bilgiler 30*20'lik bir A matrisine yerleştirilmiştir. PUT Altprogramı bu matrisi kullanarak devrenin şeklini çizer. Eğer A matrisindeki bütün sayısal değerler 0 ise matriste bilgi yok anlamındadır. Yani A matrisindeki sayısal değerleri kullanarak ekrana devrenin şeklini çizen PUT Altprogramı, bütün değerler 0 olduğu için ekrana birşey çizemez. Devredeki herbir değişim, matrisin değişmesi demek olduğundan devredeki herbir değişimde PUT Altprogramı tekrar çalışarak önceki devre şeklini silip yeni devre şeklini çizer. Çıktı hesaplanmasında program önce 30*20'lik A matrisindeki sayısal değerleri ve devre elemanlarıyla ilgili kuralları kullanarak 30*200'lük B matrisine yerleştirmek amacıyla yeni sayısal değerleri oluşturarak bu değerleri B matrisine yerleştirir. Sonra B matrisindeki sayısal değerleri iki tane 1*3000'lik matris yardımıyla çıktı hesaplanmasında kullanmaktadır.

2. SAYISAL MANTIK DEVRELERİ

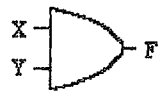
Kapı, Flip-Flop, Toplayıcı, Kod çözücü, Kodlayıcı, Bilgi seçici, Bilgi dağıtıcı, Kaydedici gibi devre elemanları kullanılarak oluşturulan devrelere sayısal mantık devreleri denir. Temel devre elemanları kapılardır. Diğer devre elemanları kapılar kullanılarak oluşturulabilir. Sayısal mantık devrelerinde pozitif mantık ve negatif mantık olmak üzere iki temel mantık vardır. Pozitif mantıkta '1' daha pozitif gerilim değerini, negatif mantıkta ise '1' daha negatif gerilim değerini tanımlar. Sayısal mantık devrelerinde genellikle 0 ve +5 Volt gibi iki gerilim seviyesi ile pozitif mantık kullanılmaktadır. Gerilim 0 Volt ise '0' ile, +5 Volt ise '1' ile tanımlanır (Bayram, 1989).

2.1. Kapılar

Transistör ve dirençler yardımıyla oluşturulan, girdilerinin durumuna göre çıkışında '1' yada '0' değerleri üreten temel sayısal devre elemanlarına kapı denir. Üç tanesi temel, dört tanesi bu temel kapılar yardımıyla oluşturulan kapılar olmak üzere yedi tane kapı vardır. AND, OR, NOT kapıları temel kapılardır. NAND, NOR, EXOR, EXNOR kapıları ise temel kapı yapıları kullanılarak oluşturulan kapılardır. Bu yedi kapının hepsi de tek çıkışa sahiptir. Not kapısı tek girişli, diğer altı kapı ise en az iki girişlidir (Bayram, 1989).

2.1.1. AND kapısı

Şekil 2.1'de iki girişli bir AND kapısı ve doğruluk tablosu verilmiştir. AND kapısının çıktısının aldığı değer girdilerin birbirleriyle çarpımıdır. Yani sadece bütün girdi değerleri '1' olduğunda çıktı '1', diğer durumlarda çıktı '0' olur. İki tane iki girişli AND kapısından birinin çıkışı diğerinin girişlerinden birine bağlanırsa üç girişli bir AND kapısı elde edilebilir (Biswas, 1993; Wiatrowski et al., 1980).



X	Y	F
0	0	0
0	1	0
1	0	0
1	1	1

Şekil 2.1. İki girişli bir AND kapısı ve doğruluk tablosu

2.1.2. OR kapısı

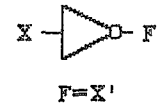
Şekil 2.2'de iki girişli bir OR kapısı ve doğruluk tablosu verilmiştir. OR kapısının en az girdilerinden bir tanesi '1' ise çıktısı '1', diğer durumlarda çıktısı '0' dır. İki tane iki girişli OR kapısından birinin çıkışı diğerinin girişlerinden birine bağlanırsa üç girişli bir OR kapısı elde edilebilir (Biswas, 1993; Wiatrowski et al., 1980).

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

Şekil 2.2. İki girişli bir OR kapısı ve doğruluk tablosu

2.1.3. NOT kapısı

Şekil 2.3'de bir NOT kapısı ve doğruluk tablosu verilmiştir. NOT kapısı tek girişe ve tek çıkışa sahiptir. NOT kapısının çıktısının aldığı değer, girdisinin aldığı değerin tersidir. Yani girdi '0' ise çıktı '1', girdi '1' ise çıktı '0' dır (Biswas, 1993; Wiatrowski et al., 1980).

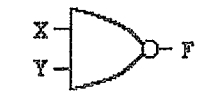


X	F
0	1
1	0

Şekil 2.3. Bir NOT kapısı ve doğruluk tablosu

2.1.4. NAND kapısı

Şekil 2.4'de iki girişli bir NAND kapısı ve doğruluk tablosu verilmiştir. AND kapısının çıkışına bir NOT kapısı bağlanarak NAND kapısı elde edilebilir. NAND kapısının çıktısının aldığı değer girdilerin birbirleriyle çarpımından elde edilen değerin tersidir (Biswas, 1993; Wiatrowski et al., 1980).



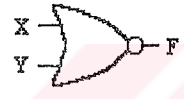
F=(X.Y)'=X'+Y'

X	Y	F
0	0	1
0	1	1
1	0	1
1	1	0

Şekil 2.4. İki girişli bir NAND kapısı ve doğruluk tablosu

2.1.5. NOR kapısı

Şekil 2.5'de iki girişli bir NOR kapısı ve doğruluk tablosu verilmiştir. OR kapısının çıktısına bir NOT kapısı bağlanarak NOR kapısı elde edilebilir. NOR kapısının bütün girdileri '0' ise çıktısı '1', diğer durumlarda çıktı '0' dır (Biswas, 1993; Wiatrowski et al., 1980).



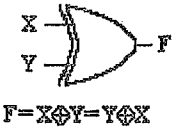
F=(X+Y)'=X'.Y'

X	Y	F
0	0	1
0	1	0
1	0	0
1	1	0

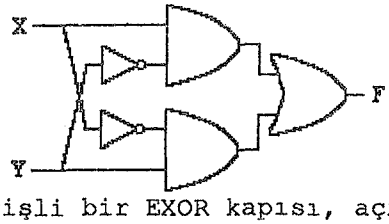
Şekil 2.5. İki girişli bir NOR kapısı ve doğruluk tablosu

2.1.6. EXOR kapısı

Şekil 2.6'da iki girişli bir EXOR kapısı, açık gösterimi ve doğruluk tablosu verilmiştir. İki girişli bir EXOR kapısı için girdiler birbirlerine eşit ise çıktı '0', birbirlerinden farklı ise çıktı '1' dir. İki tane iki girişli EXOR kapısından birinin çıkışı diğerinin girişlerinden birine bağlanırsa üç girişli bir EXOR kapısı elde edilebilir (Biswas, 1993; Wiatrowski et al., 1980).



F=X⊕Y=Y⊕X
=X.Y'+X'.Y

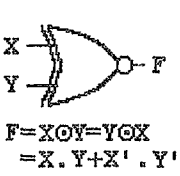


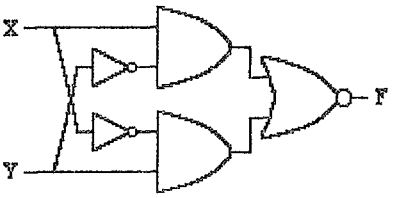
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

Şekil 2.6. İki girişli bir EXOR kapısı, açık gösterimi ve doğruluk tablosu

2.1.6. EXNOR kapısı

Şekil 2.7'de iki girişli bir EXNOR kapısı, açık gösterimi ve doğruluk tablosu verilmiştir. İki girişli bir EXNOR kapısı için girdiler birbirlerine eşit ise çıktı '1', birbirlerinden farklı ise çıktı '0' dır. EXOR kapısının çıkışına bir NOT kapısı bağlanarak bir EXNOR kapısı elde edilebilir. İki tane iki girişli EXNOR kapısından birinin çıkışı diğerinin girişlerinden birine bağlamakla üç girişli bir EXNOR kapısı elde edilebilir (Biswas, 1993; Wiatrowski et al., 1980).



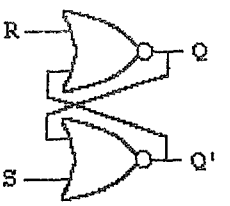


X	Y	F
0	0	1
0	1	0
1	0	0
1	1	1

Şekil 2.7. İki girişli bir EXNOR kapısı, açık gösterimi ve doğruluk tablosu

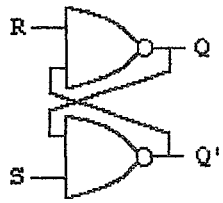
2.2. Flip-Floplar

Şekil 2.8 ve şekil 2.9'da biri NOR kapılarıyla diğeri NAND kapılarıyla kurulan iki temel Flip-Flop devre ve doğruluk tabloları verilmiştir. JK Flip-Flop, RS Flip-Flop, D Flip-Flop, T Flip-Flop gibi Flip-Floplar bir temel Flip-Flop devreye bir kaç Kapı, bir uç Clock-Pluse(CP) jenaratörünün bağlanacağı uç olacak şekilde bırakılarak, uygun bir şekilde bağlanmasıyla elde edilebilir. Şekil 2.10'da Clock-Pulse jenaratörü tarafından üretilen bir CP sinyali verilmektedir (Mano, 1984).



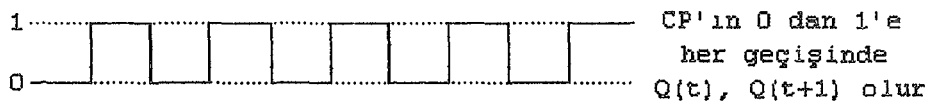
S	R	Q
0	0	Q
0	1	0
1	0	1
1	1	İzin yok

Şekil 2.8. NOR kapılarıyla kurulan temel Flip-Flop devre ve doğruluk tablosu



S	R	Q
0	0	İzin yok
0	1	0
1	0	1
1	1	Q

Şekil 2.9. NAND kapılarıyla kurulan temel Flip-Flop devre ve doğruluk tablosu



Şekil 2.10. Clock-Pulse jenaratörü tarafından üretilen CP sinyali

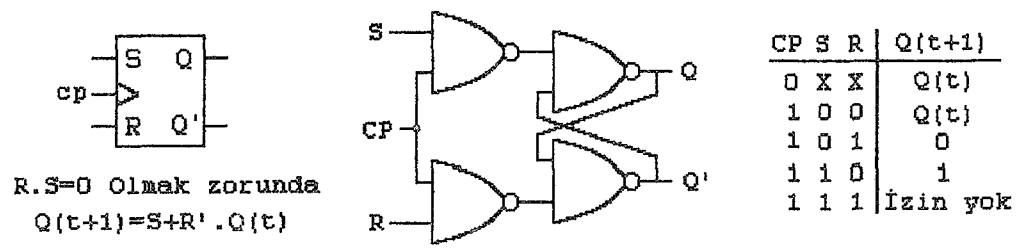
Bütün Flip-Flopların Q çıktısı '0' ise Q' çıktısı '1', Q çıktısı '1' ise Q' çıktısı '0' dır. Bir Flip-Flopun CP'sine bir CP darbesi verilmezse girdiler ne olursa olsun çıktılarında değişme olmaz. Yani girdilerindeki değişme eğer bir CP darbesi yoksa çıktılarını etkileyemez. Bundan dolayı Flip-Floplar ikili sayıları depolamak için kullanılabilir (Bartee, 1991; Mano, 1984).

2.2.1. RS Flip-Flop

Şekil 2.11'de bir RS Flip-Flop, açık gösterimi ve doğruluk tablosu verilmiştir. RS Flip-Flop'un girdilerinden ikisine birden '1' verilmez. Verilirse ve CP darbesi de varsa çıktıların ikisi birden '1' olur. Bu durum doğruluk tablosu kuralını bozar (Mano, 1984; Biswas, 1993).

Her CP darbesi verilmesi durumunda aşağıdaki şartlar tekrarlanmaktadır:

1) S=0, R=0 ise Q değişmez.
2) S=0, R=1 ise Q =0 olur.
3) S=1, R=0 ise Q =1 olur.

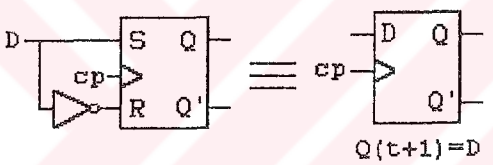


CP	S	R	Q(t+1)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	0
1	1	0	1
1	1	1	İzin yok

Şekil 2.11. Bir RS Flip-Flop, açık gösterimi ve doğruluk tablosu

2.2.2. D Flip-Flop

Şekil 2.12'de bir D Flip-Flop ve doğruluk tablosu verilmiştir. D Flip-Flop, şekil 2.12'de görüldüğü gibi bir NOT kapısı yardımıyla RS Flip-Flop'tan rahatlıkla elde edilebilir. NOT kapısı S ve R'ın ikisinin birden 0 veye 1 olmasını önler. Bir CP darbesi verilmesi durumunda Q çıktısı girdinin değerini alır. Her CP darbesinde aynı durum tekrarlanmaktadır (Biswas, 1993).



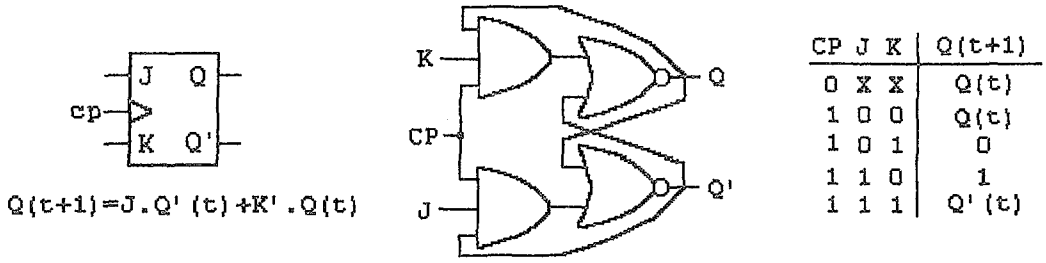
CP	D	Q(t+1)
0	X	Q(t)
1	0	0
1	1	1

Şekil 2.12. Bir D Flip-Flop ve doğruluk tablosu

2.2.3. JK Flip-Flop

Şekil 2.13'de bir JK Flip-Flop, açık gösterimi ve doğruluk tablosu verilmiştir. Her CP darbesi verilmesi durumunda aşağıdaki durumlar tekrarlanmaktadır (Mano, 1984; Biswas, 1993):

1) J=0, K=0 ise Q değişmez.
2) J=0, K=1 ise Q = 0 olur.
3) J=1, K=0 ise Q = 1 olur.
4) J=1, K=1 ise Q = Q' olur.

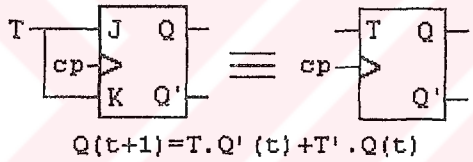


CP	J	K	Q(t+1)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	0
1	1	0	1
1	1	1	Q'(t)

Şekil 2.13. Bir JK Flip-Flop, açık gösterimi ve doğruluk tablosu

2.2.4. T Flip-Flop

Şekil 2.14'de bir T Flip-Flop ve doğruluk tablosu verilmiştir. T Flip-Flop, şekil 2.14'de görüldüğü gibi JK Flip-Flop'un iki girişi birbirine bağlanmasıyla rahatlıkla elde edilebilir. Bir CP darbesi verilmesi durumunda T=0 ise Q çıktısı değişmez, T=1 ise Q çıktısı CP darbesi verilmeden önceki değerinin tersini alır. Her CP darbesinde aynı durumlar tekrarlanmaktadır (Biswas, 1993).

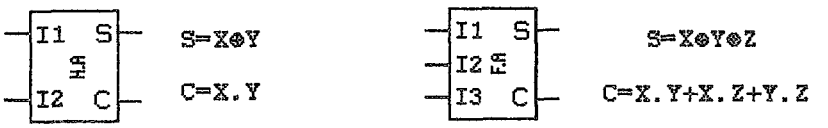


CP	T	Q(t+1)
0	X	Q(t)
1	0	Q(t)
1	1	Q'(t)

Şekil 2.14. Bir T Flip-Flop ve doğruluk tablosu

2.3. Toplayıcılar (Adders)

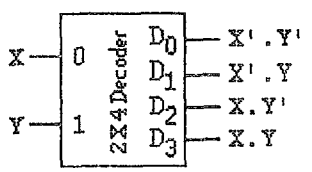
Sayısal mantık devrelerinde kullanılan iki temel toplayıcı tipi vardır. Bunlar Yarım Toplayıcı ve Tam Toplayıcı olmak üzere ikiye ayrılır. Şekil 2.15'de bir Yarım Toplayıcı ve bir Tam Toplayıcı verilmiştir. Yarım Toplayıcı ikilik düzendeki iki tane tek bit'lik sayıyı toplar. Tam Toplayıcı ikilik düzendeki üç tane tek bit'lik sayıyı toplar. Tam Toplayıcının Yarım Toplayıcıdan farkı, birden fazla bit'lik bir toplayıcının daha kolay bir şekilde yapılmasına imkan sağlamasıdır. Mesela ikilik düzende N bit'lik bir toplayıcı yapılacak ise N tane Tam Toplayıcı kulanılarak birinciden N'inciye kadar bir öncekinin C çıktısı bir sonrakinin herhangi bir girişine bağlanmaktadır (Bartee, 1991; Tinder, 1991).



Şekil 2.15. Bir Yarım Toplayıcı ve bir Tam Toplayıcı

2.4. Kod Çözücüler (Decoders)

İkili düzende belirtilen bilgilerin anlaşılan klasik şekle dönüştürülmesini sağlayan devrelere Kod Çözücü adı verilmektedir. Bir Kod Çözücü N tane girişe sahipse 2^N tane çıkışı vardır. Şekil 2.16'da 2*4'lük bir Kod Çözücü ve doğruluk tablosu verilmiştir (Bayram, 1989).

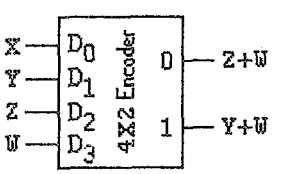


X	Y	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Şekil 2.16. 2*4'lük bir Kod Çözücü ve doğruluk tablosu

2.5. Kodlayıcılar (Encoder)

Kodlayıcılar, Kod Çözücülerin tersi işlem yapar; yani bilinen klasik şekildeki bilgileri sayısal mantık devrelerinin işlem yapabileceği şekle dönüştürür. Bir Kodlayıcı 2^N girişe sahipse N tane çıkışı vardır. Şekil 2.17'de 4*2'lik bir Kodlayıcı ve doğruluk tablosu verilmiştir (Bayram, 1989).

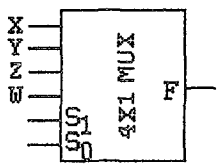


X	Y	Z	W	D_0	D_1
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Şekil 2.17. 4*2'lik bir Kodlayıcı ve doğruluk tablosu

2.6. Bilgi Seçiciler (Multiplexers)

Birden fazla girişteki sayısal bilgiyi, kontrol girişlerine bağlı olarak belirli bir sıra içinde tek bir çıkış hattına aktaran devrelerdir. Bütün Bilgi Seçicilerin tek çıkışı vardır. 2^M 'e N denilirse N*1'lik bir Bilgi Seçicinin M tane kontrol girişi, N tane normal girişi ve 1 tane çıkışı vardır. Şekil 2.18'de 4*1'lik bir Bilgi Seçici ve doğruluk tablosu verilmiştir (Mano, 1984).



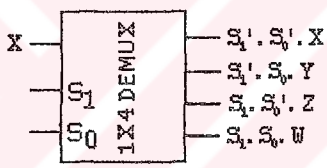
S_1	S_0	F
0	0	X
0	1	Y
1	0	Z
1	1	W

$$F=S_1'.S_0'.X+S_1'.S_0.Y+S_1.S_0'.Z+S_1.S_0.W$$

Şekil 2.18. 4*1'lik bir Bilgi Seçici ve doğruluk tablosu

2.7. Bilgi Dağıtıcılar (Demultiplexers)

Bilgi Dağıtıcılar, Bilgi Seçicilerin tersi işlem yapar; yani bir girişten ardarda gelen sayısal bilgileri kontrol girişlerine bağlı olarak istenilen hatta sıra ile dağıtan devrelerdir. Bütün Bilgi Dağıtıcıların 1 tane normal girişi vardır. 2^M 'e N denilirse 1*N'lik bir Bilgi Dagıtıcının M tane kontrol girişi, N tane çıkışı ve 1 tane normal girişi vardır. Şekil 2.19'da 1*4'lük bir Bilgi Dağıtıcı ve doğruluk tablosu verilmiştir (Mano, 1984).

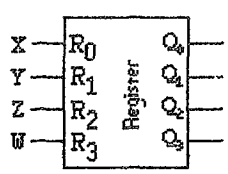


S_1	S_0	D_0	D_1	D_2	D_3
0	0	X	0	0	0
0	1	0	X	0	0
1	0	0	0	X	0
1	1	0	0	0	X

Şekil 2.19. 1*4'lük bir Bilgi Dağıtıcı ve doğruluk tablosu

2.7. Kaydediciler (Registers)

Kaydediciler, Flip-Flop ve kapılar kullanılarak oluşturulan devre elemanlarıdır. Şekil 2.20'de 4 bit'lik bir Kaydedici verilmiştir. Kaydediciler, bilgisayarda bilgi depolamada, ikili toplayıcı ve çıkarıcılarda bilgi tutmada ve transferinde kullanılır. Ayrıca Çarpma ve bölme, ikili sayıları sağa ve sola kaydırmanın bir şekli olduğundan sağa ve sola kaydırmalı kaydedicilerle çarpma ve bölme işlemlerinde de kullanılabilir (Tinder, 1991).



$$Q_0(t+1)=X$$
$$Q_1(t+1)=Y$$
$$Q_2(t+1)=Z$$
$$Q_3(t+1)=W$$

Şekil 2.20. 4 bit'lik bir Kaydedici

3. PASCAL PROGRAMLAMA DİLİ

Bilgisayarların herhangi bir işi yapması için verilen komutlar dizisine program, programın yazılıp bilgisayara verilmesi ve çalıştırılmasına programlama denir. Bilgisayarlar kendilerine ne yapmaları gerektiği doğru komutlarla verilirse, çeşitli türde problemlerin çözümünü elde edebilir. Bilgisayarların anlayacağı dil makina dilidir. Fakat bu dille program yazmak zor olduğu için BASIC, FORTRAN, COBOL, PASCAL, C gibi yüksek seviyeli diller geliştirilmiştir. Bu dillerden herhangi biri ile yazılmış bir program, derleninceye yani makina diline dönüştürülünceye kadar, bilgisayar tarafından doğrudan işletilmez. Bu işlem derleyici denilen bir bilgisayar programıyla yapılabilmektedir (Eren vd., 1992).

PASCAL programlama dili, matematiksel çalışmalar için geliştirilmiş olup en yeni dillerden biridir. Yüksek seviyeli dillerden olduğu için bu dil ile yazılmış bir programın doğrudan işletilmesi için PASCAL programlama dilini anlayacak bir derleyici tarafından derlenmesi lazımdır (Bayburan, 1990).

PASCAL programlama dilinde yazılmış bir programın genel şekli aşağıdaki gibidir:

```
PROGRAM İsmi;
Tanımlama Bloğu;
BEGIN
 İcra Blogu;
END.
```

3.1. Veri Tipleri

Veri tipleri, tanımlama bloğunda kullanılan verilerin tiplerini belirlemek için kullanılır. Standart ve özel olmak üzere ikiye ayrılır. Standart veri tipleri programcı tarafından ayrıca tanımlanmasına gerek olmayan veri tipleridir. Bunlar Integer (Tamsayı), Real (Gercel sayı), Byte (0 ile 255 arasındaki tamsayıları içerir), Char (Karekter), String (Birden fazla karakterden olur),

Boolean (False yani Yanlış ve True yani Doğru şeklinde iki değişken içerir) olmak üzere altı tanedir. Özel veri tipleri ise TYPE deyimi yardımıyla tip tanımlama bloğunda programcı tarafından tanımlanan veri tipleridir. Aşağıda standart ve özel veri tipleriyle ilgili bir örnek verilmiştir (Bayburan, 1990).

```
TYPE günler=(pzt,salı,çarş,perş,cuma,cts,pazar);
VAR X,Y:Integer;gün:günler;Ch:Char;İsim:String;
```

3.2. Tanımlama Bloğu

İcra bloğunda kullanılan parametrelerin tanımlama bloğunda tanımlanması gerekir. Programcı tarafından kullanılan

a) Etiket tanımlama bloğu
b) Sabit tanımlama bloğu
c) Değişken tanımlama bloğu
d) Tip tanımlama bloğu
e) Fonksiyon tanımlama bloğu
f) Prosedür tanımlama bloğu

olmak üzere 6 ayrı tanımlama bloğu vardır (Bayburan, 1990).

3.2.1. Etiket tanımlama bloğu

Etiketler, LABEL deyimi yardımıyla etiket tanımlama bloğu adı verilen blokta tanımlanır. Program içindeki bazı adresleri tanımlamak için kullanılır. GOTO deyimiyle birlikte program içindeki akış yönünü değiştirmeye yarar. Bu bloğun örnek gösterimi aşağıdaki gibidir (Bayburan, 1990):

```
LABEL Etiket1,Etiket2,.....,EtiketN;

Örnek: Label ilk,son,10,20,30,a,b;
```

3.2.2. Sabit tanımlama bloğu

Program içerisinde defalarca tekrar edilen birtakım sabitlerin değerleri yerine bu sabit değerleri temsil eden simgelerin kullanılması, programın anlaşılabilirliğini artırır ve bu sabit değerler değiştirilmek istenildiğinde kolaylık sağlar. Sabitler CONST deyimi yardımıyla sabit tanımlama bloğu adı verilen blokta tanımlanır. Bu bloğun örneği aşağıda verilmiştir (Bayburan, 1990):

```
CONST Sabit1=Değer1;Sabit2=Değer2;.....;SabitN=DeğerN;
```

```
Örnek: Const PI=3.14;Harf='A';İsim='Hasan';A=100;
```

3.2.3. Değişken tanımlama bloğu

Bir TURBO PASCAL programında kullanılacak bütün değişkenlerin önceden tanımlanması gerekir. Değişkenler VAR deyimi yardımıyla değişken tanımlama bloğu adı verilen blokta tanımlanır. Bu bloğun örneği aşağıda verilmiştir (Bayburan, 1990):

```
VAR Değişken1:Veritipi1;Değişken2:Veritipi2;
    Değişken3:Veritipi3;....;DeğişkenN:VeritipiN;
```

```
Örnek: Var X,Y,Z:Integer;Miktar:Real;Ch:Char;
           Soru:Boolean;Kelime:String[10];
```

3.2.4. Tip tanımlama bloğu

Tipler, TYPE deyimi yardımıyla tip tanımlama bloğu adı verilen blokta tanımlanır. PASCAL'da programcının standart veri tiplerine bağlı kalma zorunluluğunu ortadan kaldırır. Programcı, standart veri tiplerinin yanı sıra kendi özel veri tiplerini oluşturma ve bu tipleri kullanma imkanına sahip olur. Bunlar Basit, Dizisel, Kayıtlama, Küme ve İşaretleyici özel veri tipleridir. Bu bloğun örnek gösterimi aşağıda verilmiştir (Bayburan, 1990):

```
TYPE TipİsmiTanımlama;
```

3.2.5. Fonksiyon tanımlama bloğu

Fonksiyonlar, bir veya birden fazla değer üzerinde işlem yaparak bir tek değerin üretilmesi amacıyla tasarlanmış altprogramlardır. Fonksiyon ismi prosedür ismi gibi kendi başına bir komut olarak kullanılamaz. Ancak bir ifade içerisinde veya bir komut cümlesi içinde operant olarak yer alabilir. Bu bloğun örneği aşağıda verilmiştir (Bayburan, 1990):

```
FUNCTION İsim(Parametre tanımları):Fonksiyonun veri tipi;
Tanım bloğu;
BEGIN
 İşlem bloğu;
END;
```

3.2.6. Prosedür tanımlama bloğu

Prosedür, aynı amaca yönelik bir dizi komut içeren bir altprogramdır. Prosedürler anaprogramın tüm işlevlerini yerine getirebilir. Bu bloğun örneği aşağıda verilmiştir (Bayburan, 1990):

```
PROCEDURE İsim(Parametre tanımları);
Tanım bloğu;
BEGIN
 İşlem bloğu;
END;
```

3.3. Aritmetik Operatörler

Aritmetik operatörler, genelde kullanılan toplama(+),çıkarma(-), çarpma(*), bölme(/) gibi işlemlerin yanı sıra, iki tamsayının bölünmesiyle elde edilen kalanın(mod) veya bölümün(div) bulunması gibi işlemleri içerir (Eskicioğlu, 1988).

3.4. Atama İşlemi

Bir değişkenin içeriğini veya herhangi bir sabiti bir değişkene atamak için kullanılır. Mesela A:=5 5'i A değişkenine atar, M:=K-10 ise K değişkeninin on eksiğini M değişkenine atar (Eskicioğlu, 1988).

3.5. Arşiv Fonksiyonları

ABS(X) : X'in mutlak değerini alır.

SQR(X) : X'in karesini alır.

SQRT(X) : X'in karekökünü alır.

EXP(X) : X'in exponensiyelini alır.

LN(X) : X'in dogal logaritmasını alır.

ODD(X) : X tek sayı ise True, çift sayı ise False değerini alır.

ROUND(X) : X'e enyakın tamsayının değerini alır.

TRUNC(X) : X'in tamsayı kısmını alır.

FRAC(X) : X'in real kısmını alır.

COS(X) : X'in kosinüsünü alır.

SIN(X) : X'in sinüsünü alır.

ARCTAN(X) : X'in arktanjantını alır.

UPCASE(CH) : CH küçük harf ise onu büyük harf yapar.

3.6. Karakter İşlemleri

KEYPRESSED : Klavyeden bir tuşa basılmışsa true değerini, basılmamışsa false değerini alır.

READKEY : Klavyeden girilen karakteri okur. Okuma işleminden sonra KEYPRESSED'i false yapar.

IF KEYPRESSED THEN CH:=READKEY : KEYPRESSED true ise basılan karakteri CH değişkenine atar ve KEYPRESSED'i false yapar;

3.7. Akış Deyimleri Ve Döngüler

Normal şartlar altında bir TURBO PASCAL programında değerlendirme programın ilk deyiminden başlamakta ve son deyimine kadar birbirini izleyip devam etmektedir. Ancak bu akış araya girecek akış deyimleri ve döngülerle saptırılabilmektedir (Eren vd., 1992).

3.7.1. GOTO deyimi

GOTO deyimi ile birlikte kullanılan etiket ismi, program içinde işlem akışının yönlendirilebileceği adresi belirler. GOTO deyimiyle kullanılan bu etiket isimlerinin programın etiket tanımlama bloğunda tanımlanması gerekir. GOTO deyiminin örneği aşağıda verişlmiştir (Eren vd., 1992):

```
GOTO etiket ismi;
```

3.7.2. IF deyimi

Bir şartın doğru veya yanlış olmasına bagli olarak belli program parçalarının icra görmesini veya görmemesini sağlar. Bu deyimin örnek gösterimi

```
IF Şart THEN Blok;
```

veya

```
IF Şart THEN Blok1 ELSE Blok2;
```

şeklindedir (Eren vd., 1992).

Örnek : If x>5 then y:=x+y else x:=x+1;

3.7.3. CASE OF deyimi

Bir değişkenin birden fazla değer ile karşılaştırılmasını yapan ve bir eşitliğin bulunması halinde belli program parçalarının icrasını sağlayan deyimdir. Bu deyimin örneği aşağıda verilmiştir (Eren vd., 1992):

```
CASE Değişken OF
 Etiket1:Blok1;Etiket2:Blok2;.....;EtiketN:BlokN;
END;
```

```
Örnek : Case harf of
          'A':Altprogram1; 'B':Altprogram2; 'C':Altprogram3;
        End;
```

3.7.4. FOR döngüsü

Belli bir program parçasının icrasının üst üste tekrarlanması amacıyla kullanılabilmektedir. Bu döngünün örneği aşağıda verilmiştir (Eren vd., 1992):

```
FOR Değişken:=İlkDeğer TO SonDeğer DO Blok;
```

```
Örnek : For X:=1 to 100 do
        Begin
         If X=1 then Toplam:=0;
         Toplam:=Toplam+X;
        End;
```

3.7.5. REPEAT döngüsü

Bir program bloğunun belirtilen şartlar sağlanıncaya kadar üst üste icrasını sağlar. Bu döngünün örneği aşağıda verilmiştir (Eren vd., 1992):

```
REPEAT Blok UNTIL Şart;
```

```
Örnek : Repeat A:=A+1 until A>15;
```

3.7.6. WHILE döngüsü

Belli bir program bloğunun belirtilen şartlar sağlandığı sürece üst üste icrasını sağlar. Bu döngünün örneği aşağıda verilmiştir (Eren vd., 1992):

```
WHILE Şart DO Blok;

Örnek : While (A<=10)or(C>=50) do begin A:=A+B;C:=A+C end;
```

3.8. Diziler

Bir değişkene ikinci bir değer atandığında var olan eski değer silinmektedir. Değişkene verilen değerlerin silinmesi istenmiyorsa değişkeni dizi olarak tanımlamak gerekir. Diziler tek boyutlu veya daha fazla boyutlu olabilir. Tek boyutlu bir dizi PASCAL programlama dilinde aşağıdaki gibi tanımlanabilir (Eskicioğlu, 1988).

```
VAR DiziAdı:Array[AltLimit..ÜstLimit] of VeriTipi;
```

Mesela aynı özelliklere sahip birden fazla tek boyotlu dizi tanımlamak istenirse aşağıdaki gibi tanımlanır.

```
TYPE DiziAdı=Array[AltLimit..ÜstLimit] of VeriTipi;
VAR DiziAdı1,DiziAdı2,DiziAdı3:DiziAdı;
```

İstenirse daha fazla boyutlu diziler tanımlanabilir. Aşağıda dizi adı A, veri tipi tamsayı, alt limitleri 1, üst limitleri M, N, K değişkenleri olan üç boyutlu bir dizi tanımlanmıştır.

```
VAR A:Array[1..M,1..N,1..K] of Integer;
```

3.9. Dosyalama

Read/readln deyimleri yardımıyla klavyeden girilen veri giriş hızı çok yavaş olduğundan fazla sayıda verinin girilmesine ihtiyac duyulması durumunda bilgisayarın etkin kullanımı engellenebilecektir.

Bu engel verilerin manyetik şerit, disk, disket gibi ortamlarda programdan ayrı bir yerde saklanmasıyla giderilebilir. Bu işleme dosyalama işlemi denir. Böylece, verilerin tümü yada bir kısmı istenildiği zaman değişik amaçlar için ayrı ayrı programlar tarafından kullanılabilir (Kay, 1985).

Veri dosyaları; sıralı erişimli dosyalar, rasgele erişimli dosyalar ve indekslenmiş erişimli dosyalar olmak üzere üç tanedir. Sayısal mantık devreleriyle ilgili programda sıralı erişimli dosyalar kullanıldığı için sadece sıralı erişimli dosyalar anlatılacaktır (Kay, 1985).

3.9.1. Sıralı erişimli dosyalar

Herbir kayıt diğerini izlemekte olup, kayıtlar erişim sıralı olmaktadır. İstenilen bir kayda erişmek için o kayıttan önceki tüm kayıtlar okunacağından bir zaman kaybı söz konusu olmaktadır. Fakat dosya tasarımının basit oluşu, kayıtların manyetik ortamda boşluk bırakmaksızın birbirlerini izlemeleri sonucu kayıt ortamının en etkin biçimde kullanılması gibi bazı üstünlükleri vardır (Kay, 1985).

Sıralı erişimli dosyalarda erişimin gerçekleşmesi için önce dosyanın açılması gerekir. Reset, Rewrite, Append komutları dosya açmak için kullanılır. Dosyadan kayıt okunacaksa Reset, dosya ilk defa oluşturularak yeni kayıt yapılacaksa Rewrite, mevcut bir dosyaya kayıt ilavesinde bulunulacaksa Append komutu kullanılarak açılır. Reset ile açılmış sıralı dosyalarda sadece Read ve Readln komutları çalıştırılabilir. Rewrite veya Append ile açılmış dosyalarda sadece Write ve Writeln komutları çalıştırılabilir. Sıralı dosyaları temsil eden dosya değişkenleri TEXT tanımlaması ile değişken tanımlama bloğunda tanımlanabilir. Mesela F ile temsil edilen bir dosya, VAR F:TEXT; şeklindedir (Bayburan, 1990; Eren vd., 1992).

ASSIGN(F,'Örnek.dat') : Bir ismin dosya değişkenine aktarılması için kullanılır (Bayburan, 1990; Eren vd., 1992).

RESET(F) : F ile temsil edilen ve diskte mevcut olan dosyayı okuma modunda açar. Diğer bir deyişle, okuyucu kafa dosyanın ilk kaydı üzerinde konumlanır. F ile temsil edilen dosya diskte mevcut değilse bir run-time hatası meydana gelir (Bayburan, 1990; Eren vd., 1992).

REWRITE(F) : F ile temsil edilen dosyayı diskte ilk defa oluşturmak için kullanılır. Eğer bu dosya diskte mevcut ise içindeki bilgiler silinerek yeniden oluşturulur (Bayburan, 1990; Eren vd., 1992).

APPEND(F) : F ile temsil edilen ve diskte önceden bulunan dosyayı yazma modunda açar ve yazıcı kafayı dosyanın sonuna konumlandırır. Böylece dosyanın sonundan itibaren dosyaya kayıt ilavesi gerçekleştirilebilir. F ile temsil edilen dosya diskte mevcut değilse bir run-time hatası meydana gelir (Bayburan, 1990; Eren vd., 1992).

CLOSE(F) : F ile temsil edilen dosyayı kapar. Kayıt işlemleri sonrasında Close deyimi kullanılmazsa, yapılan kayıtlar geçersiz olur (Bayburan, 1990; Eren vd., 1992).

EOF(F) : Okuyucu kafa, F ile temsil edilen dosyanın sonunda ise true, aksi taktirde false değeri verir. Reset ile açılmış bir dosya üzerinden kayıt okunurken, her okuma işlemi öncesinde bu komut kullanılarak okunabilecek bir bilgi kaydının mevcut bulunup bulunmadığı kontrol edilmelidir (Bayburan, 1990; Eren vd., 1992).

WRITE(F,D1,D2,..,DN) ve WRITELN(F,D1,D2,..,DN) : D1,D2,..,DN değerleri, F ile temsil edilen dosyaya yazılır (Bayburan, 1990; Eren vd., 1992).

READ(F,D1,D2,..,DN) ve READLN(F,D1,D2,..,DN) : D1,D2,..,DN değerleri, F ile temsil edilen dosyadan okunacak değişkenlerdir (Bayburan, 1990; Eren vd., 1992).

3.10. Grafik Ekran Ve Mouse

TURBO PASCAL'da grafik ekranda çalışmak için öncelikle metin ekranın grafik ekrana dönüştürülmesi gerekir. Ekranı grafik ekran için düzenleyen ve grafik ekranda grafik çizmek için kullanılan komutları içinde bulunduran dosya GRAPH.TPU dosyasıdır. Bundan dolayı programın en başına 'USES GRAPH' komutunu yazmak gerekir. Program sonunda metin ekrana dönebilmek için CLOSEGRAPH komutu kullanılır. Mouse'u devreye sokan ve mouse ile ilgili komutları içinde barındıran dosya DRIVERS.TPU dosyasıdır. Programda mouse kullanılmak istenirse programın en başına USES DRIVERS komutunu yazmak gerekir. Aşağıda bu işleri yapabilecek bir programın örnek şekli verilmiştir (O'Brien, 1983):

```
USES GRAPH,DRIVERS;
VAR GD,GM:INTEGER;
BEGIN
 GD:DETECT;
 INITGRAPH(GD,GM,'');
 IF GRAPHRESULT<>GROK THEN HALT(1);
 INITEVENTS;
 .
 .
 CLOSEGRAPH;
END.
```

INITEVENTS : Programda Mouse'u devreye sokar ve Mouse okunu gösterir.

HIDEMOUSE : Mouse okunu saklar.

SHOWMOUSE : Mouse okunu gösterir.

A:=8*MOUSEWHERE.X; B:=8*MOUSEWHERE.Y : Mouse okunun bulunduğu yerin X kordinatını A'ya, Y kordinatını B'ye yükler.

REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING : Mouse'un tuşlarından birine basılana kadar bekler.

REPEAT UNTIL MOUSEBUTTONS=MBLEFTBUTTON : Mouse'un sol tuşuna basılana kadar bekler.

REPEAT UNTIL MOUSEBUTTONS=MBRIGHTBUTTON : Mouse'un sağ tuşuna basılana kadar bekler.

SETVIEWPORT(X1,Y1,X2,Y2,TRUE) : Ekranın içinde sol üst köşesi (X1,Y1) noktası, sağ alt köşesi (X2,Y2) noktası olan bir dikdörtgen alan alıp onu normal ekran olarak kabul eder. Dışarda kalan alanı kullanmaz. Normal ekran olarak kabul edilen bölgenin X kordinatı 0 ile X2-X1, Y kordinatı 0 ile Y2-Y1 arasında değişir.

CLEARVIEWPORT : Normal ekran olarak kabul edilen bölgeyi temizlemek için kullanılır.

EXIT : Bilgisayar EXIT komutu okuduğunda duruma göre anaprogramdan, fonksiyondan veya altprogramdan dışarı çıkılır.

CIRCLE(X, Y, R) : Merkezi (X,Y) noktası, yarıçapı R olan bir daire çizer.

ELLİPSE(X, Y, AÇI1, AÇI2, XR, YR) : X kordinatındaki yarıçapı XR, Y kordinatındaki yarıçapı YR, başlangıç açısı AÇI1, bitiş açısı AÇI2 ve merkezi (X,Y) noktası olan eliptik bir yay çizer.

ARC(X, Y, AÇI1, AÇI2, R) : Yarıçapı R, başlangıç açısı AÇI1, bitiş açısı AÇI2 ve merkezi (X,Y) noktası olan bir yay çizer.

LINE(X1, Y1, X2, Y2) : (X1,Y1) noktasıyla (X2,Y2) noktası arasına bir hat çizer.

RECTANGLE(X1, Y1, X2, Y2) : Sol üst köşesi (X1,Y1) noktası, sağ alt köşesi (X2,Y2) noktası olan bir dikdörtgen çizer.

PUTPIXEL(X, Y, COLOR) : (X,Y) noktasının bulunduğu yere rengi COLOR olan bir nokta koyar.

SETCOLOR(COLOR) : Çizilecek şeklin rengi bu komutla ayarlanır.

FLOODFILL(X ,Y , COLOR) : (X,Y) noktasını içine alan çevresi COLOR rengiyle çizili en küçük alanın içini COLOR rengine boyar.

OUTTEXTXY(X, Y, ' ') : (X,Y) noktasından itibaren yazı yazar.

SETFILLSTYLE(PATTERN, COLOR) : Boyanacak alanın içinin şeklini ve rengini belirler.

SETTEXTSTYLE(FONT, DIRECTION, CHARSIZE) : Yazılacak yazının fontunu, yönünü ve büyüklüğünü belirler.

4. SAYISAL MANTIK DEVRELERİ İLE İLGİLİ BENZETİM PROGRAMI

Bu benzetim programı PASCAL dili kullanılarak yazılmış olup temel mantığı matrisler üzerine kurulmuştur. 30*20'lik A matrisi temel matris olup başlangıçta bütün değişkenlerinin değerleri 0 dır. Bunun manası, matriste bilgi yok demektir. Yani A matrisindeki sayısal değerleri kullanarak önce ekranını temizleyip sonra ekrana devre şekli çizen PUT Altprogramı, A matrisindeki bütün değerler 0 olduğu için ekrana birşey çizemez. A matrisinin her değişmesinde PUT altprogramı tekrar çalışır. Ekranda bir devre oluşturulmak istenildiğinde istenilen devre tamamlanana kadar ekrandaki devre şekli devamlı değişir. Devrenin değişmesi demek A matrisin değişmesi demek olduğundan devredeki değişmeler aslında PUT Altprogramının tekrar tekrar çalışmasından elde edilmektedir. A matrisi devamlı değiştiğinden PUT Altprogramının çizdiği devre şekli de devamlı değişir.

AND, NAND, OR, NOR, EXOR, EXNOR ve NOT kapıları, JK, D, T ve RS Flip-Flopları, Yarım Toplayıcı, Tam Toplayıcı, 1*2'lik Kod Çözücü, 2*4'lük Kod Çözücü, 2*1'lik Kodlayıcı, 4*2'lik Kodlayıcı, 2*1'lik Bilgi Seçici, 4*1'lik Bilgi Seçici, 1*2'lik Bilgi Dağıtıcı, 1*4'lük Bilgi Dağıtıcı, 2 bit'lik Kaydedici, 4 bit'lik Kaydedici gibi devre elemanlarının benzetimlerinin herbiri için ayrı ayrı toplam 23 tane altprogram yazılmıştır. Bu Altprogramların isimleri AND_KAPISI, NAND_KAPISI, OR_KAPISI, NOR_KAPISI, EXOR KAPISI, EXNOR_KAPISI, NOT_KAPISI, JKFF, DFF, TFF, RSFF, HALF_ADDER, FULL_ADDER, DECODER12, DECODER24, ENCODER21, ENCODER42, MULTIPLEXER21, MULTIPLEXER41, DEMULTIPLEXER12, DEMULTIPLEXER14, REGISTER2, REGISTER4 dır.

Devre elemanlarının yerleştirilmesi, iptal edilmesi veya yerlerinin değiştirilmesi, aralarına hatların döşenmesi veya iptal edilmesi, girişlerine girdilerin yerleştirilmesi veya iptal edilmesi, istenilen devre elemanının istenilen çıktısının hesaplanması, ekrandaki devrenin silinmesi, programdan çıkılması, çizilen devrenin kaydedilmesi veya kaydedilmiş bir devrenin geri çağrılması PUT_CIRCUIT_ELEMENT, TAKE_CIRCUIT_ELEMENT, MOVE_CIRCUIT_ELEMENT, PUT_WIRE, TAKE_WIRE, PUT_INPUT, TAKE_INPUT, FIND_OUTPUT, NEW, GOOUT,

SAVE_OR_TAKE Altprogramları vasıtasıyla yapılır. Bu Altprogramlar mouse yardımıyla şu şekilde çağrılır. Mouse oku ekrandaki menüde üzerlerinde Put_Circuit_Element, TakeE, MoveE, PutW, TakeW, NEW, EXIT, PutI, TakeI, FindO, SAVE ve FILES yazılı pencerelerden birinin üzerine getirilip sağ veya sol tuşuna basılırsa program gereğince T'nin önceki değeri silinir ve yerine Tablo 4.1'deki değeri alır. Program gereğince T=1 ise PUT_CIRCUIT_ELEMENT, T=2 ise TAKE_CIRCUIT_ELEMENT, T=3 ise MOVE_CIRCUIT_ELEMENT, T=4 ise PUT_WIRE, T=5 ise TAKE_WIRE, T=6 ise NEW, T=7 ise GOOUT (EXIT), T=8 ise PUT_INPUT, T=9 ise TAKE_INPUT, T=10 ise FIND_OUTPUT, T=12 veya T=13 ise SAVE_OR_TAKE Altprogramı çalışır.

Tablo 4.1. T'deki değişme

Pencere üzerindeki yazı	T'nin alacağı yeni değer
Put_Circuit_Element	1
TakeE	2
MoveE	3
PutW	4
TakeW	5
NEW	6
EXIT	7
PutI	8
TakeI	9
FindO	10
SAVE	12
FILES	13

30 satır ve 20 sütundan oluşan A matrisinde herbir devre elemanı için bir satırlık yer ayrılmıştır. Herbir satırın 1. sütunu devre elemanının X kordinatını, 2. sütunu devre elemanının Y kordinatını, 3. sütünu ise devre elemanının hangi devre elemanı olduğunu içerir. Devre elmanlarının girişlerine dışarıdan girilen X, Y, Z, W, 0, 1 gibi girdi değerleri için program gereğince matristeki yerlerine X için 31, Y için 32, Z için 33, W için 34, 0 için 35, 1 için 36 yazılır.

Herhangi bir satırın bütün sütunları 0 ise o satır boştur. Herhangi bir satırın 3. sütununun değeri

1 ise o satır bir AND Kapısının
2 ise o satır bir NAND Kapısının
3 ise o satır bir OR Kapısının
4 ise o satır bir NOR Kapısının
5 ise o satır bir EXOR Kapısının
6 ise o satır bir EXNOR Kapısının
7 ise o satır NOT Kapısının
8 ise o satır bir JK Flip-Flop'un
9 ise o satır bir D Flip-Flop'un
10 ise o satır bir T Flip-Flop'un
11 ise o satır bir RS Flip-Flop'un
12 ise o satır bir Yarım Toplayıcının
13 ise o satır bir Tam Toplayıcının
14 ise o satır 2*1'lik bir Kodlayıcının
15 ise o satır 4*2'lik bir Kodlayıcının
16 ise o satır 2*1'lik bir Bilgi Seçicinin
17 ise o satır 4*1'lik bir Bilgi Seçicinin
18 ise o satır 1*2'lik bir Bilgi Dağıtıcının
19 ise o satır 1*4'lük bir Bilgi Dağıtıcının
20 ise o satır 1*2'lik bir Kod Çözücünün
21 ise o satır 2*4'lük bir Kod Çözücünün
22 ise o satır 2 bit'lik bir Kaydedicinin
23 ise o satır 4 bit'lik bir Kaydedicinin

konumuyla, giriş ve çıkışlarıyla ilgili bilgileri içerir.

Herhangi bir satırın 3. sütununun sayısal değeri 1 ile 7 arasında ise şu şartlar sağlanmalıdır. 4. sütunun sayısal değeri devre elemanlarının çıkışları ile bu devre elemanının girişleri arasındaki bağlantı hatların sayısına eşittir. Eğer varsa 5., 7., 9., 11. sütunlar bu devre elemanlarının satır numaralarını, 6., 8., 10., 12. sütunlar bağlantı hatlarının bu devre elemanlarının hangi çıkışından geldiğinin bilgisini içerir. 13. sütunun sayısal değeri bu devre elemanının girişlerine dışarıdan girilen girdi sayısına eşittir. 14., 15., 16., 17. sütunlar ise eğer varsa bu girdilerin sayısal gösterim değerlerini ihtiva eder(X için 31, Y için 32 gibi).

Herhangi bir satırın 3. sütununun sayısal değeri 8 ile 23 arasında ise şu şartlar sağlanmalıdır. 4., 6., 8., 10., 12., 14. sütunlar eğer varsa çıkışlarından bu devre elemanının girişlerine bağlantı hatları bulunan devre elemanlarının satır numaralarını veya eğer varsa dışarıdan bu devre elemanının girişlerine girilen girdilerin sayısal gösterim değerlerini içerir. 5., 7., 9., 11., 13., 15. sütunlar ise eğer varsa çıkışlarından bu devre elemanının girişlerine bağlantı hatları bulunan devre elemanlarından gelen bu bağlantı hatlarının hangi çıkışından geldiğinin bilgisini ihtiva eder.

4.1. Programda Kullanılan Önemli Altprogramlar

DIGITAL_LOGIC_CIRCUITS Altprogramı : Diğer bütün altprogramları kullanan bir altprogramdır.

KONUM Altprogramı : Devreyi yukarı, aşağı, sağa, sola kaydırma esnasında kordinatların nerede olduğunu belirler.

HELP Altprogramı : Programı kullanma esnasında programı kullanan kişiye yardımcı olur.

BOXES0 Altprogramı : Üzerinde ana altprogramların kısaltılmış isimleri yazılı olan kutuların şeklini ekranın yukarısına çizer.

RENK0 Altprogramı : Üzerlerinde ana altprogramların kısaltılmış isimleri yazılı olan kutulardan istenileni diğerlerinden ayırmak için değişik bir renge boyamak için kullanılır.

BOXES1 Altprogramı : Üzerlerinde devre elemanlarının isimleri yazılı olan kutuların şeklini ekranın yukarısına çizer.

RENK1 Altprogramı : Üzerlerinde devre elemanlarının isimleri yazılı olan kutulardan istenileni diğerlerinden ayırmak için değişik bir renge boyamak için kullanılır.

MARK Altprogramı : İstenilen herhangi bir devre elemanını veya o devre elemanının istenilen herhangi bir girişini veya herhangi bir çıkışını işaretlemek için kullanılır.

CIRCUIT_ELEMENT Altprogramı : A matrisinin i. satırındaki sayısal değerleri kullanarak i. satırın 3. sütununun temsil ettiği devre elemanını i. satırın 1. ve 2. sütunlarının gösterdiği yere çizer.

LINES Altprogramı : A matrisinin i. satırındaki sayısal değerleri kullanarak varsa devre elemanlarının çıkışlarıyla i. satırın temsil ettiği devre elemanının girişleri arasındaki bağlantı hatlarını çizer.

INPUTS Altprogramı : A matrisinin i. sütunundaki sayısal değerleri kullanarak varsa i. satırın temsil ettiği devre elemanının girişlerine dışardan girilen girdileri yerleştirir.

PUT Altprogramı : Önce ekranı temizleyip sonra CIRCUIT_ELEMENT, LINES, INPUTS gibi bazı altprogramları ve A matrisindeki sayısal değerleri kullanarak ekrana A matrisinin temsil ettiği devre şeklini çizer.

NEW Altprogramı : A matrisinin bütün sayısal değerlerini 0 yapıp PUT Altprogramını çalıştırır. Önce ekranı temizleyip sonra A matrisindeki sayısal değerleri kullanarak ekrana devre şekli çizen PUT Altprogramı, A matrisinin bütün sayısal değerleri 0 olduğu için hiçbir devre şekli çizemediğinden sadece ekrani temizlemiş olur. Yani NEW Altprogramının amacı ekranı temizleyip yeni bir devre çizmek için hazır duruma getirmektir.

GOOUT Altprogramı : Programdan çıkmak için kullanılır.

PUT_CIRCUIT_ELEMENT Altprogramı : Devre elemanlarını yerleştirmek için kullanılır.

TAKE_CIRCUIT_ELEMENT Altprogramı : İstenmeyen devre elemanlarını iptal etmek için kullanılır. Ayrıca bu altprogramda iptal edilen devre elemanlarıyla ilgili girişleri ve çıkışları da iptal eder.

MOVE_CIRCUIT_ELEMENT Altprogramı : Yerleri değiştirilmek istenilen devre elemanlarının yerlerinin değiştirilmesi için kullanılır. Ayrıca bu altprogram yerleri değiştirilen devre elemanlarına bağlı girişleri ve çıkışları da yeni duruma göre tekrar düzenler.

PUT_WIRE Altprogramı : Devre elemanları arasına hatlar yerleştirmek için kullanılır.

TAKE_WIRE Altprogramı : Devre elemanları arasındaki istenmiyen hatları iptal etmek için kullanılır.

PUT_INPUT Altprogramı : Devre elemanlarının girişlerine X, Y, Z, W, 0, 1 gibi girdiler yerleştirmek için kullanılır.

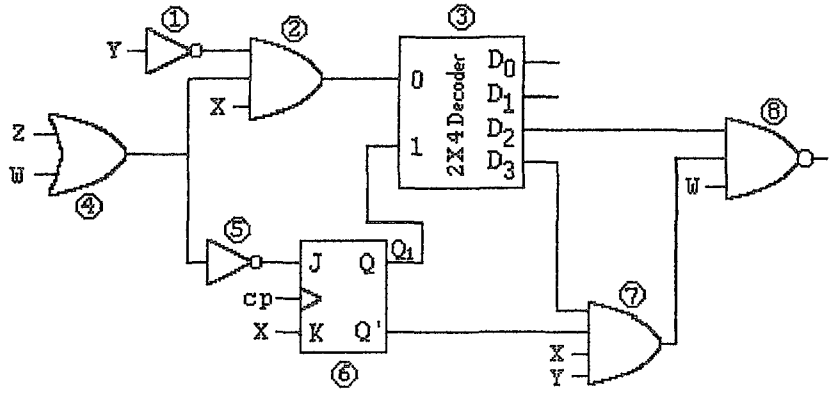
TAKE_INPUT Altprogramı : Devre elemanlarının girişlerindeki istenmeyen girdileri iptal etmek için kullanılır.

FIND_OUTPUT Altprogramı : İstenilen devre elemanlarının istenilen çıkışları üzerindeki çıktıları hesaplamak için kullanılır.

SAVE_OR_TAKE Altprogramı : Ekranda çizilen devreleri kaydetmek ve kaydedilmiş devrelerden istenileni geri çağırmak için kullanılır.

4.2. Örnek Program

Şekil 4.1'de blok diyagramı verilen Sayısal Mantık Devresinin 30*20'lik A matrisi şeklinde gösterimi Tablo 4.2'de gösterilmiştir. PUT Altprogramı, bu matristeki sayısal değerleri kullanarak şekil 4.1'de görünen devrenin şeklini çizer. Matristeki her değişiklik, devredeki her değişiklik anlamına gelir. Yapılan her değişiklikte PUT Altprogramı tekrar çalışarak önceki devre şeklini silip yeni devre şeklini çizer.



Şekil 4.1. Bir Sayısal Mantık Devresinin Blok Diyagramı

Tablo 4.2. Devrenin 30*20'lik A matrisindeki gösterimi

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1			7	0	0	0	0	0	0	0	0	0	1	32	0	0	0	0	0	0
2			1	2	1	0	4	0	0	0	0	0	1	31	0	0	0	0	0	0
3			21	2	0	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4			3	0	0	0	0	0	0	0	0	0	2	33	34	0	0	0	0	0
5			7	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6			8	5	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	1
7			1	2	3	9	6	2	0	0	0	0	2	31	32	0	0	0	0	0
8			2	2	3	8	7	0	0	0	0	0	1	34	0	0	0	0	0	0

Herhangi bir devrenin A matrisi aşağıdaki kurallar kullanılarak elde edilmektedir:

a) İki devre elemanı arasındaki bağlantı hattı i. satırın temsil ettiği devre elemanının herhangi bir girişine j. satırın temsil ettiği kapı veya 2*1'lik kodlayıcının çıkışından geliyorsa i. satırdaki yerlerine j ve 0 yazılır.

b) İki devre elemanı arasındaki bağlantı hattı i. satırın temsil ettiği devre elemanının herhangi bir girişine j. satırın temsil ettiği Flip-Flop, Yarım Toplayıcı, 1*2'lik Bilgi Dağıtıcı, 1*2'lik Kod Çözücü, 2 bit'lik Kaydedici gibi bir devre elemanının 1. çıkışından geliyorsa i. satırdaki yerlerine j ve 1, 2. çıkışından geliyorsa i. satırdaki yerlerine j ve 2 yazılır.

c) İki devre elemanı arasındaki bağlantı hattı i. satırın temsil ettiği devre elemanının herhangi bir girişine j. satırın temsil ettiği 2*1'lik veya 4*1'lik bir Bilgi Seçicinin çıkışından geliyorsa i. satırdaki yerlerine j ve 3 yazılır.

d) İki devre elemanı arasındaki bağlantı hattı i. satırın temsil ettiği devre elemanının herhangi bir girişine j. satırın temsil ettiği 4*2'lik bir Kodlayıcının 1. çıkışından geliyorsa i. satırdaki yerlerine j ve 4, 2. çıkışından geliyorsa i. satırdaki yerlerine j ve 5 yazılır.

e) İki devre elemanı arasındaki bağlantı hattı i. satırın temsil ettiği devre elemanının herhangi bir girişine j. satırın temsil ettiği 1*4'lük Bilgi dağıtıcı, 2*4'lük Kod çözücü, 4 bit'lik Kaydedici gibi bir devre elemanının 1. çıkışından geliyorsa i. satırdaki yerlerine j ve 6, 2. çıkışından geliyorsa i. satırdaki yerlerine j ve 7, 3. çıkışından geliyorsa i. satırdaki yerlerine j ve 8, 4. çıkışından geliyorsa i. satırdaki yerlerine j ve 9 yazılır.

f) i. satırın temsil ettiği devre elemanının herhangi bir girişine X girilirse i. satırdaki yerine 31, Y girilirse i. satırdaki yerine 32, Z girilirse i. satırdaki yerine 33, W girilirse i. satırdaki yerine 34, 0 girilirse i. satırdaki yerine 35, 1 girilirse i. satırdaki yerine 36 yazılır.

Çıktı hesaplanırken program öncelikle 30*200'lük B, 1*3000'lik Q1, 1*3000'lik Q2 matrislerinin bütün sayısal değerleri 0 yapar ve 30*20'lik A matrisinin değerlerini de 30*20'lik O matrisine(Tablo 4.3) yükler. Program sonra 30*20'lik O matisinde bir devre elemanı bilgisini içeren herbir satır için bir kapının bilgisini içeriyorsa o satırın 6., 8., 10., 12. sütunlarındaki, içermiyorsa o satırın 5., 7., 9., 11., 13., 15. sütunlardaki sayısal değerleri aşağıdaki gibi değiştirir. O sütunların herbiri için eğer sayısal değer 0, 1, 3, 4, 6 değerlerinden herhangi biri ise o değer silinip yerine -1 değerini, 2, 5, 7 değerlerinden herhangi biri ise o değer silinip yerine -2 değerini, 8 ise yerine -3 değerini, 9 ise yerine -4 değerini koyar. Daha sonra herbir satırının ilk 50 sütunu devre elemanlarının birinci

çıktılarına, ikinci 50 sütunu devre elemanlarının ikinci çıktılarına, üçüncü 50 sütunu devre elemanlarının üçüncü çıktılarına, dördüncü 50 sütunu devre elemanlarının dördüncü çıktılarına ayrılan 30*200'lük B matrisi aşağıdaki A, B, C şıklarındaki bilgiler kullanılarak oluşturulur.

A) 30*20'lik O matrisi

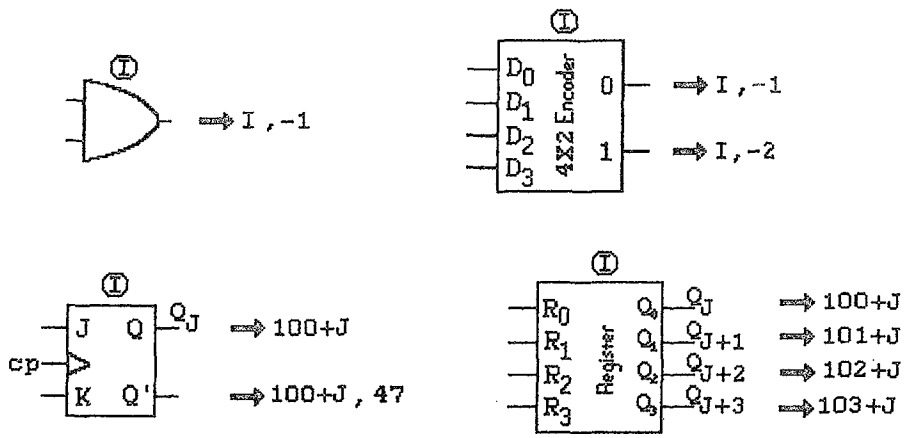
B) Devre elemanlarının çıktı fonksiyonları

C) AND{.} için 41, OR{+} için 43, EXOR{⊕} için 45, EXNOR{O} için 46, NOT{'} için 47, (için 51,) için 52, Q1 için 101, Q2 için 102, Q3 için 103,..,Q99 için 199

Tablo 4.3. 30*20'lik O matrisi

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1			7	0	0	-1	0	-1	0	-1	0	-1	1	32	0	0	0	0	0	0
2			1	2	1	-1	4	-1	0	-1	0	-1	1	31	0	0	0	0	0	0
3			21	2	-1	6	-1	0	-1	0	-1	0	-1	0	-1	0	0	0	0	0
4			3	0	0	-1	0	-1	0	-1	0	-1	2	33	34	0	0	0	0	0
5			7	1	4	-1	0	-1	0	-1	0	-1	0	0	0	0	0	0	0	0
6			8	5	-1	31	-1	0	-1	0	-1	0	-1	0	-1	0	0	0	0	1
7			1	2	3	-4	6	-2	0	-1	0	-1	2	31	32	0	0	0	0	0
8			2	2	3	-3	7	-1	0	-1	0	-1	1	34	0	0	0	0	0	0

Herhangi bir devre elemanının çıkışının 30*200'lük B matrisinde gösterimi şekil 4.2'deki ve yukardaki kurallardan yararlanılarak elde edir.



Şekil 4.2. Her hangi bir devre elemanının çıkışının 30*200'lük B matrisinde gösterimi için yararlanılacak kurallar

Bu kurallar gereğince şekil 4.1'deki devrenin 30*200'lük B matrisinde gösterimi aşağıda gösterilmiştir:

1. Satır :

İlk 50 sütun : 32,47,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

2. Satır :

İlk 50 sütun : 51,1,-1,41,4,-1,41,31,52,0,0,0,.....,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

3. Satır için

İlk 50 sütun : 51,2,-1,47,41,101,47,52,0,0,0,...,0

İkinci 50 sütun : 51,2,-1,47,41,101,52,0,0,0,...,0

üçüncü 50 sütun : 51,2,-1,41,101,47,52,0,0,0,...,0

son 50 sütun : 51,2,-1,41,101,52,0,0,0,...,0

4. Satır :

İlk 50 sütun : 51,33,43,34,52,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

5. Satır :

İlk 50 sütun : 4,47,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

6. Satır için

İlk 50 sütun : 51,51,5,-1,41,101,47,52,43,51,31,47,41,101,
52,52,0,0,0,...,0

İkinci 50 sütun : 51,51,5,-1,41,101,47,52,43,51,31,47,41,101,

52,52,47,0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

7. Satır :

İlk 50 sütun : 51,3,-4,41,6,-2,41,31,41,32,52,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

8 Satır :

İlk 50 sütun : 51,3,-3,41,7,-1,41,34,52,47,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

Diğer satırlar :

İlk 50 sütun : 60,0,0,0,...,0

İkinci 50 sütun : 0,0,0,...,0

üçüncü 50 sütun : 0,0,0,...,0

son 50 sütun : 0,0,0,...,0

dır.

Mesela A matrisinin 3. satırının temsil ettiği 2*4'lik Kod çözücünün 3. çıktısı hesaplanmak istenirse, program B matrisinin 3. satırının üçüncü 50 sütününun ilk 0 değerine kadar ki kısmını daha önce bütün değerleri 0 yapılmış olan 1*3000'lik Q1 matrisine baştan başlayarak sıra ile yükler. Q1 matrisinin durumu aşağıda gösterilmiştir.

51,2,-1,41,101,47,52,0,0,0,...,0

0'dan büyük 31'den küçük ilk eleman 2 dir. 2'den sonraki ilk eleman da -1 dir. Program 2 ve -1 değerlerini iptal eder ve araya 1*3000'lik Q2 matrisinin yardımıyla B matrisinin 2. Satırının ilk 50 sütununun ilk 0 değerine kadar ki kımını yerleştirir. Bu durumda Q1 matrisinin durumu aşağıda verilmiştir.

51,51,1,-1,41,4,-1,41,31,52,41,101,47,52,0,0,0,...,0

Program 0'dan büyük 31'den küçük eleman kalmayana kadar aynı işlemleri tekrarlar. Sonuçta aşağıdaki matris elde edilir.

51,51,32,47,41,51,33,43,34,52,41,31,52,41,101,47,52,0,0,0,...,0

Sonra yukarıda yapılan dönüştürmenin tersi işlem yapılarak sonuç bulunur. Yani 31 yerine X, 32 yerine Y, 33 yerine W, 34 yerine Z, 35 yerine 0, 36 yerine 1, 41 yerine {.}, 43 yerine {+}, 45 yerine {⊕}, 46 yerine {O}, 47 yerine {'}, 51 yerine {(}, 52 yerine {)}, 101 yerine Q1, 102 yerine Q2,..., 199 yerine Q99 kullanılır. Eğer çıktıda 0 veya 1'ler varsa program onu sadeleştirir.

5. SONUÇ VE TARTIŞMA

Bu çalışmada, sayısal mantık devrelerinin tasarımının bilgisayar benzetimiyle gerçekleştirilmesi için genel amaçlı bir yazılım Pascal Programlama Dili kullanılarak yapılmıştır. Bu yazılım programı AND, NAND, OR, NOR, EXOR, EXNOR ve NOT kapıları, JK, D, T ve RS Flip-Flopları, Yarım Toplayıcı, Tam Toplayıcı, 1*2'lik Kod Çözücü, 2*4'lük Kod Çözücü, 2*1'lik Kodlayıcı, 4*2'lik Kodlayıcı, 2*1'lik Bilgi Seçici, 4*1'lik Bilgi Seçici, 1*2'lik Bilgi Dağıtıcı, 1*4'lük Bilgi Dağıtıcı, 2 bit'lik Kaydedici, 4 bit'lik Kaydedici gibi devre elemanlarının yardımıyla oluşturulan herbir devrenin bilgisayar ekranında şeklinin rahatlıkla oluşturulmasına ve istenilen herhangi bir devre elemanının herhangi bir çıkışı üzerindeki çıktının rahatlıkla hesaplanmasına imkan sağlamaktadır. Ayrıca çizilen her devre kaydedilebilmekte ve kaydedilmiş her devre istendiğinde tekrar geri çağrılabilmektedir.

Bu program matrisler yerine işaretciler (pointers) kullanılarak yapılabilirdi. Yapılsaydı programın daha hızlı çalışması mümkün olacaktı. Fakat aynı programı yazmak için daha fazla yer gerekecek ve ayrıca yazılım mantığı da oldukca zor olacaktı.

Çizilen mantık devrelerinin yazıcıdan çıktısı alınabilecek ve her devre için doğruluk tablosu oluşturulabilecek şekilde bu benzetim programı biraz daha geliştirilecek olursa daha kullanışlı olur.

KAYNAKLAR

Atamazhori P., February 1987, Design and Implementation of logic simulator, M.Sc.Thesis, METU.

Halıcı U., 1983, An Algorithm for test set generation of combinational circuits, M.Sc.Thesis, METU.

O'Brien S.K., 1983, Turbo Pascal 6 : The Complete Reference, McGRAW-HILL, Inc.

Kay D.G., 1985, Programming for people Pascal, Mayfield Publishing Company.

Biswas N.N., 1993, Logic Design Theory, Prentice-Hall, Inc.

Wiatrowski C.A., House C.H., 1980, Logic Circuits and Microcomputer Systems, McGRAW-HILL, Inc.

Tinder R.F., 1991, Digital Engineering Design, Prentice-Hall, Inc.

Bartee T.C., 1991, Computer Architecture and Logic Design.

Mano M.M., 1984, Digital Design, Prentice-Hall, Inc.

Turbo Pascal 6.0, Library Reference, Borland International, Inc.

Turbo Pascal 6.0, Programmer's Guide, Borland International, Inc.

Turbo Pascal 6.0, User's Guide, Borland International, Inc.

Turbo Pascal 6.0, Turbo Vision Guide, Borland International, Inc.

Bayram H., 1989, Dijital Elektronik, Mapa Matbaacılık Ltd. Şti.

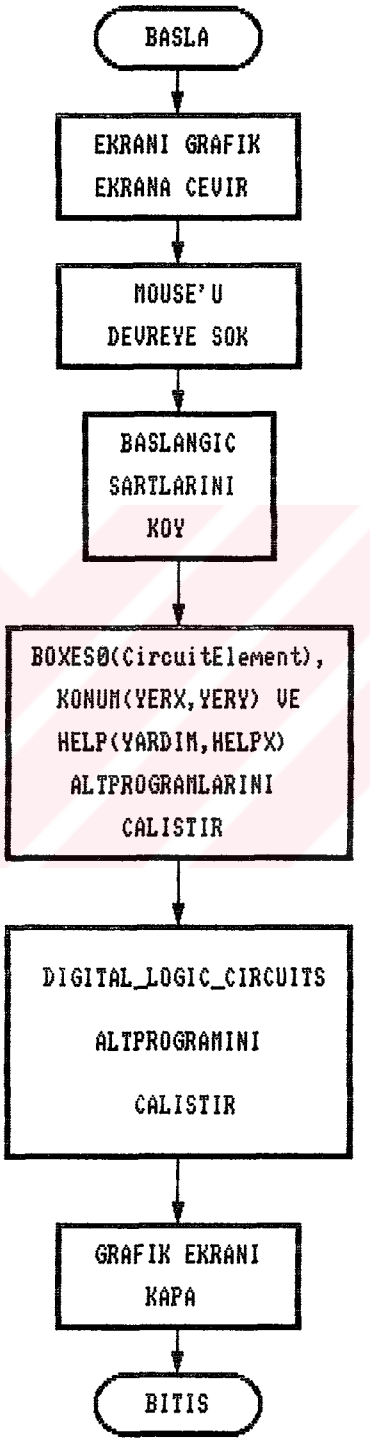
Eşkicioğlu A.M., Mart 1988, PASCAL İle Yapısal Programlama, Evrim Basım Yayım Dağıtım.

Eren Ş., Inceoğlu M., 1992, Mikrobilgisayarlar İçin Turbo Pascal 5.5, Fakülteler Kitapevi Barış Yayınları.

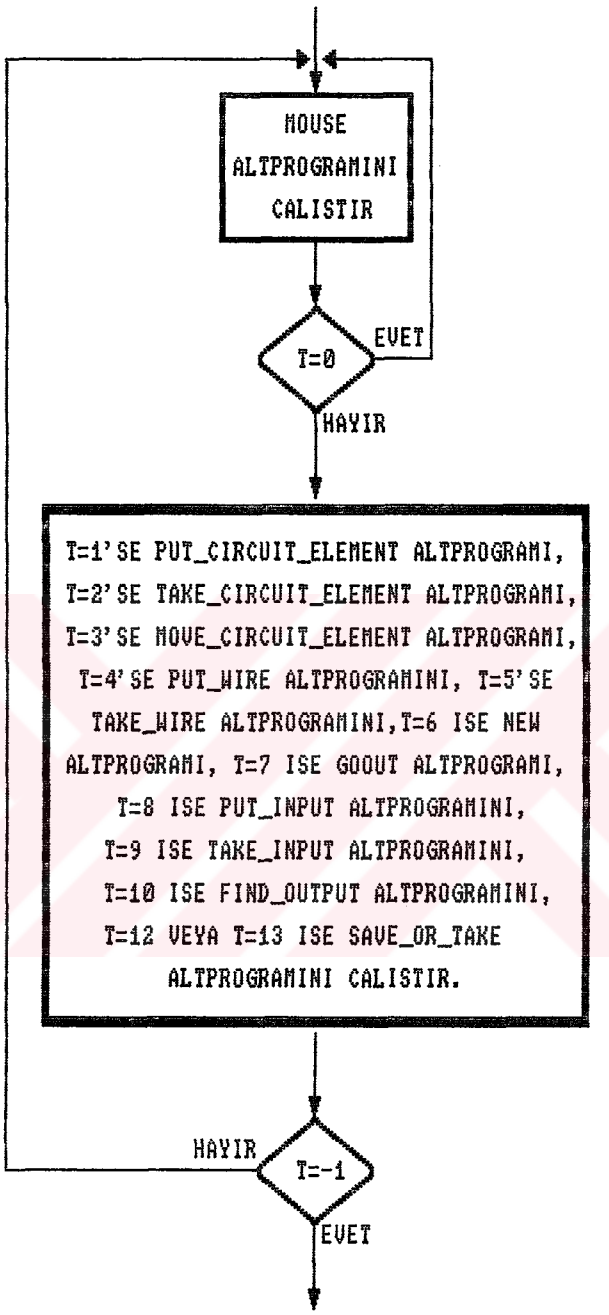
Bayburan B., 1990, Turbo Pascal, Beta Basım Yayım Dağıtım A.Ş.

EK.1. ALTPROGRAMLARIN AKIS DIYAGRAMLARI

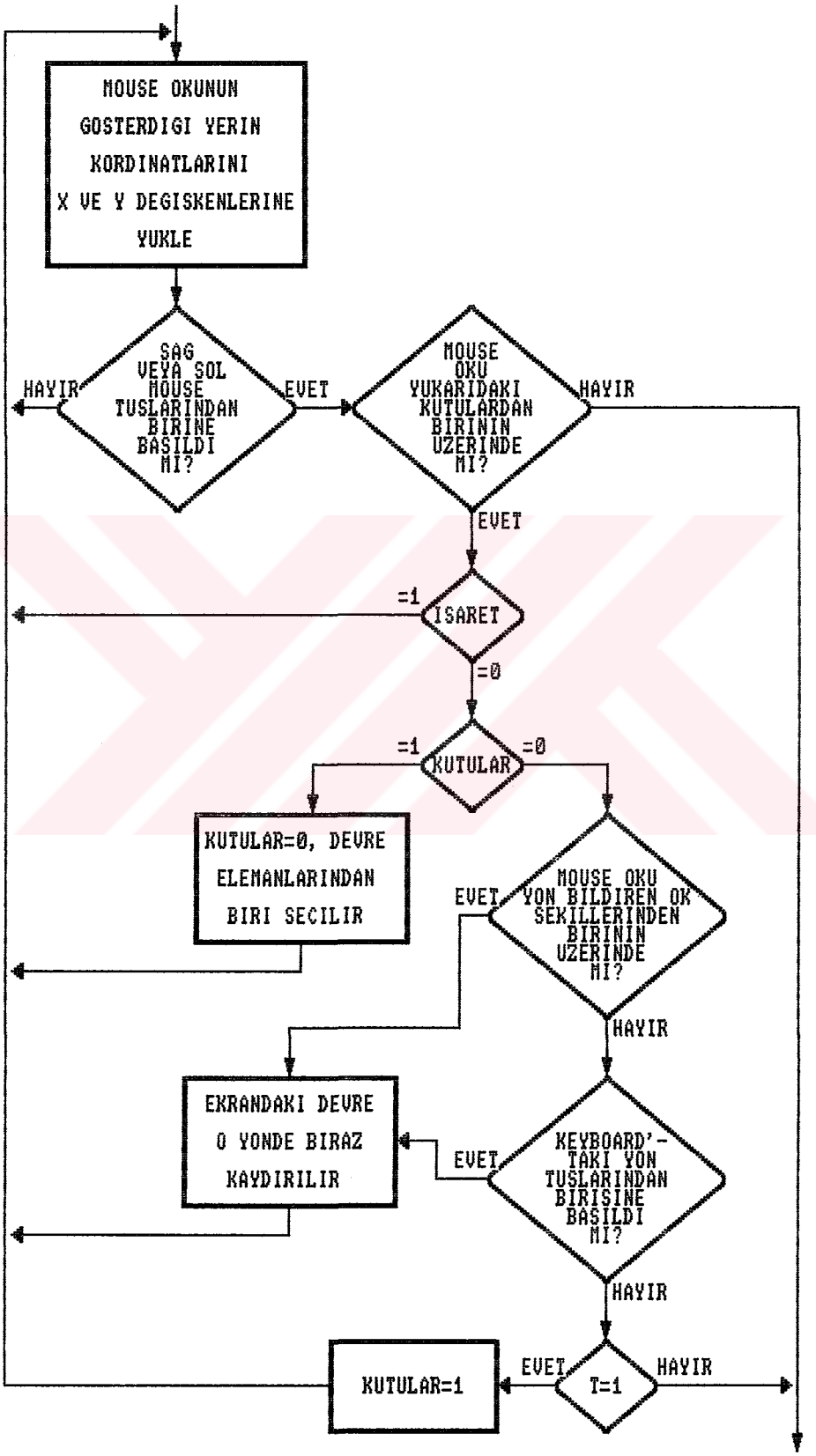
ANAPROGRAMIN AKIS DIYAGRAMI



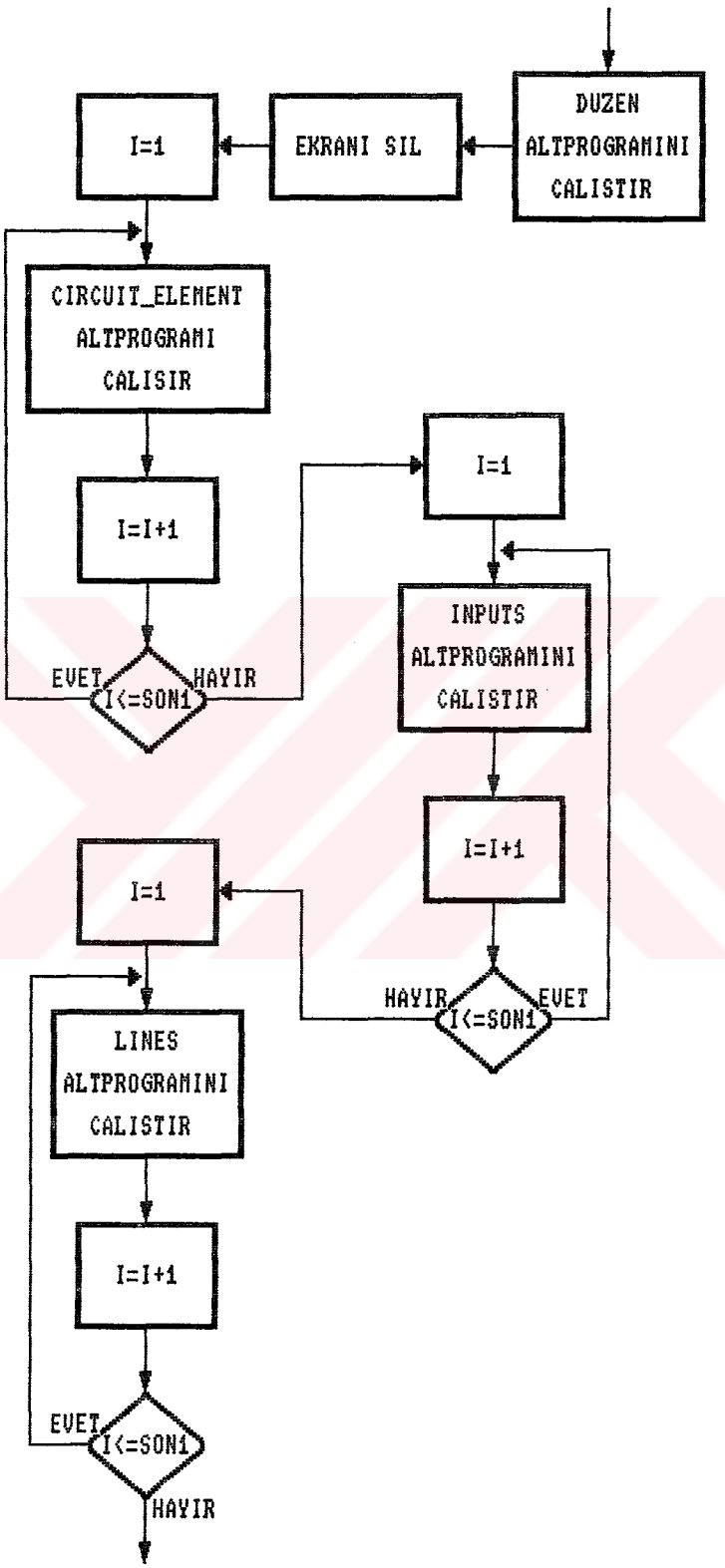
DIGITAL_LOGIC_CIRCUITS ALTPROGRAMININ AKIS DIYAGRAMI



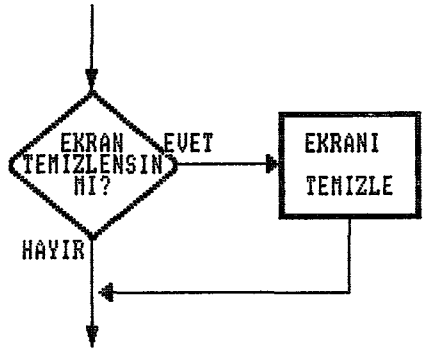
MOUSE ALTPROGRAMININ AKIS DIYAGRAMI



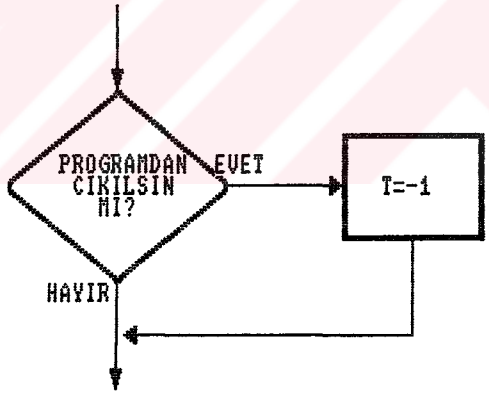
PUT ALTPROGRAMININ AKIS DIYAGRAMI



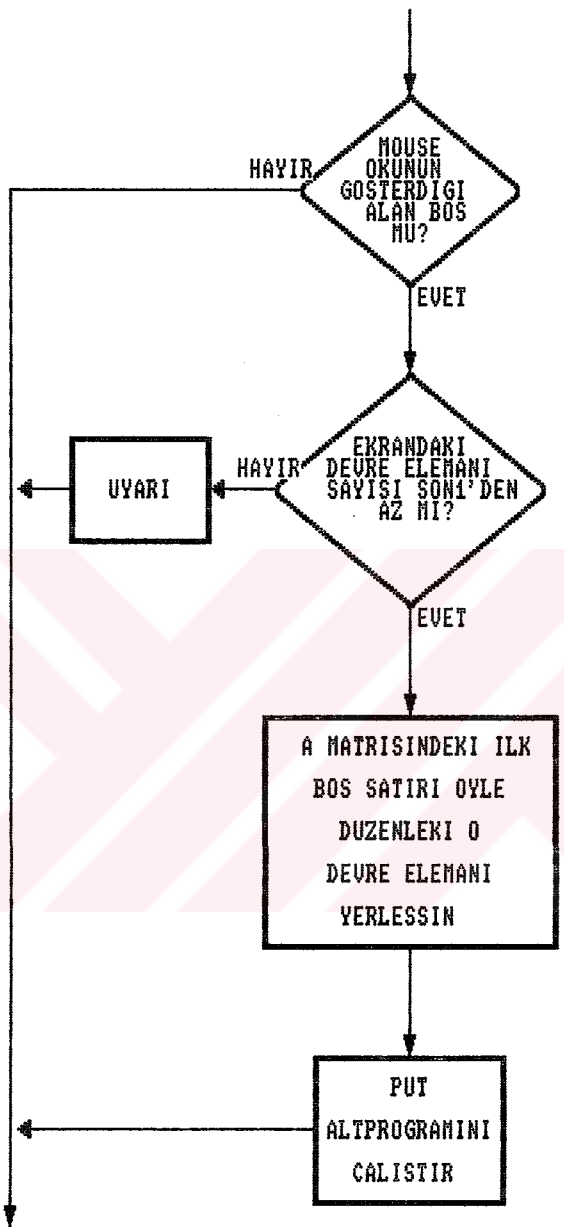
NEW ALTPROGRAMININ AKIS DIYAGRAMI



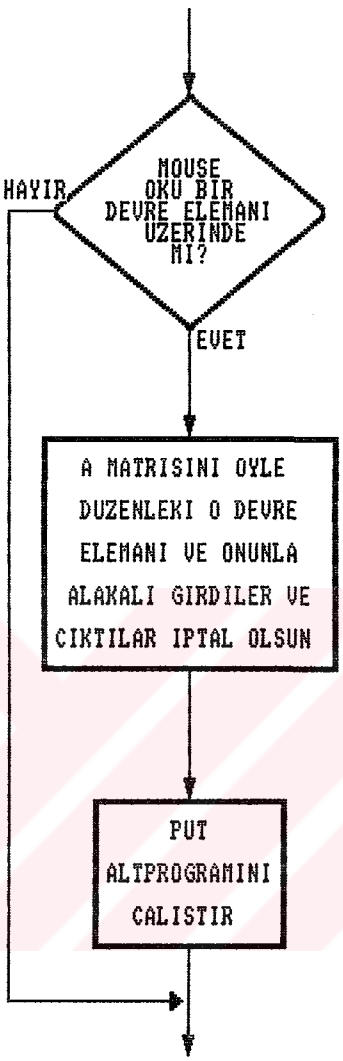
GOOUT ALTPROGRAMININ AKIS DIYAGRAMI



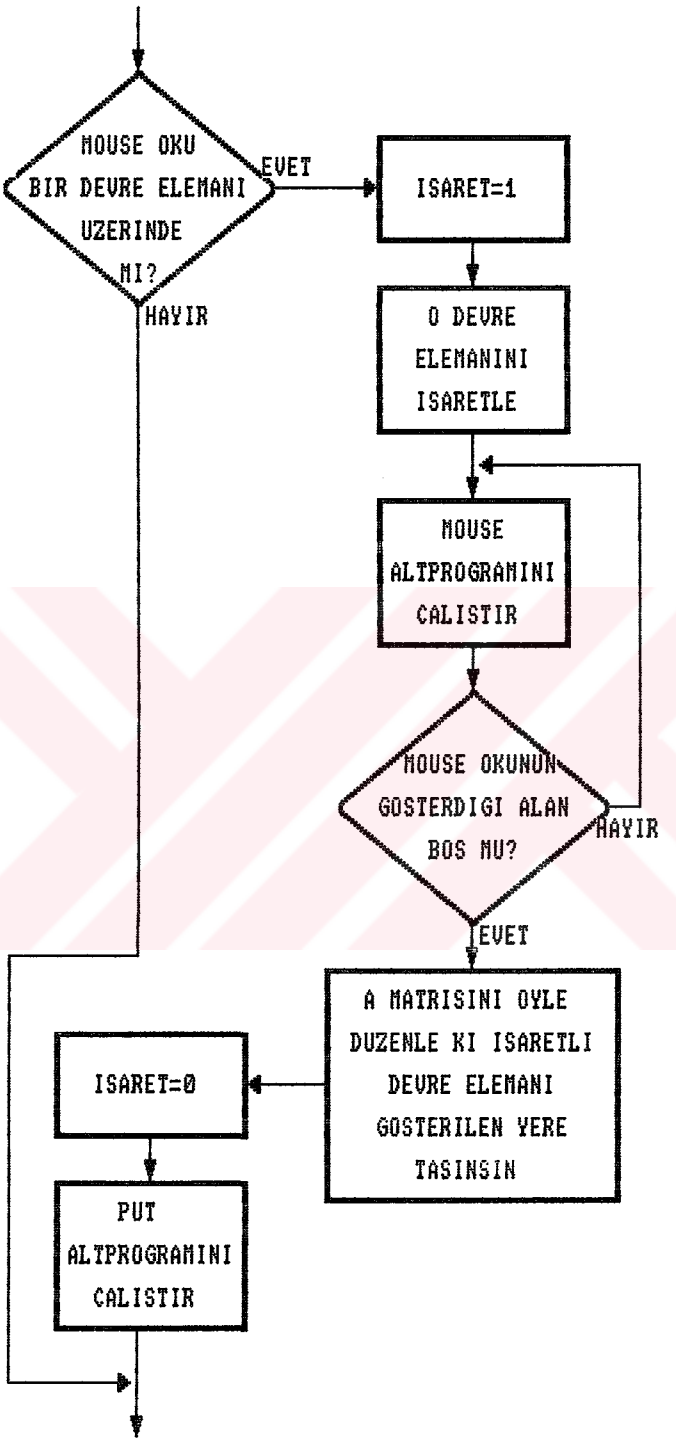
PUT_CIRCUIT_ELEMENT ALTPROGRAMININ AKIS DIYAGRAMI



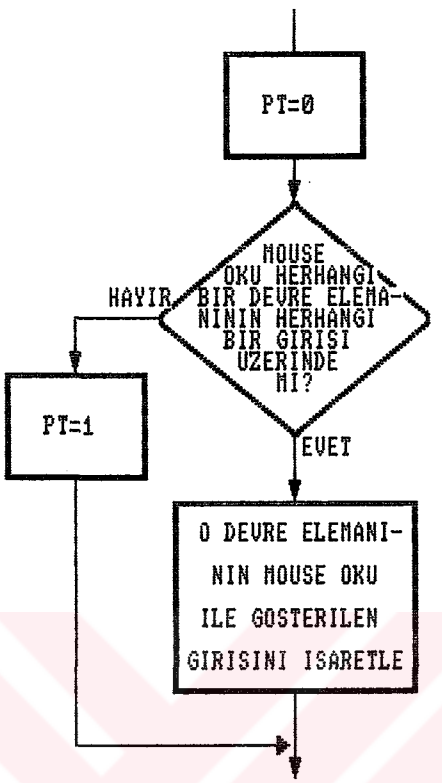
TAKE_CIRCUIT_ELEMENT ALTPROGRAMININ AKIS DIYAGRAMI



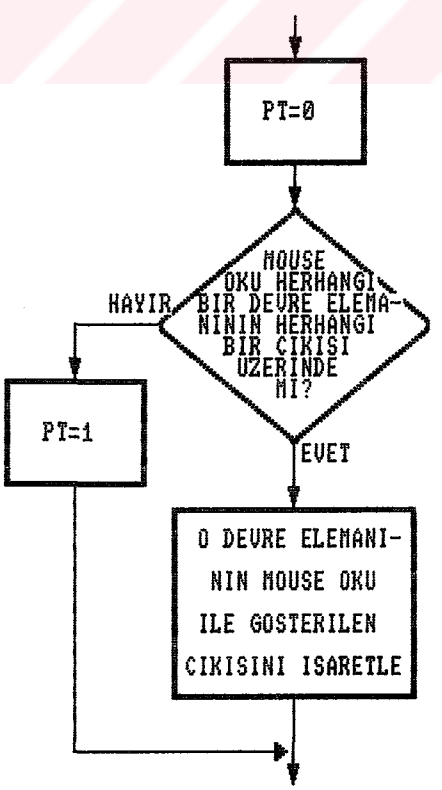
MOVE_CIRCUIT_ELEMENT ALTPROGRAMININ AKIS DIYAGRAMI



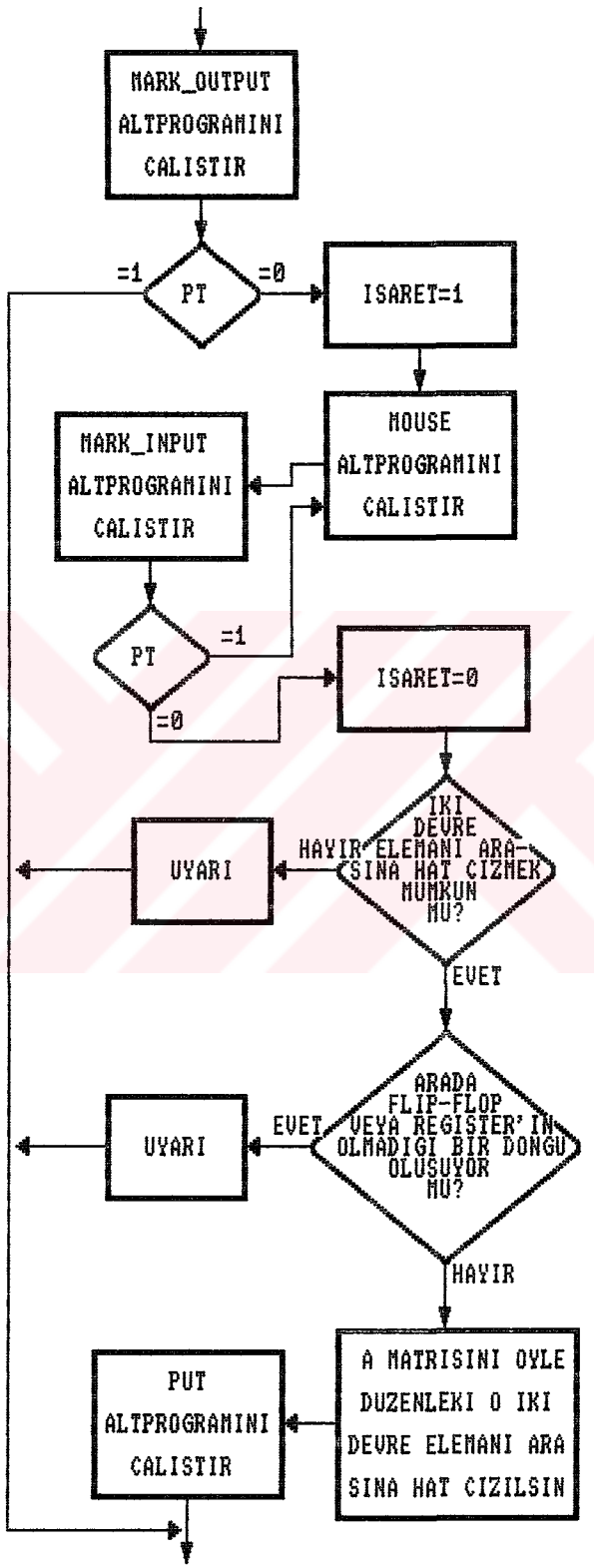
MARK_INPUT ALTPROGRAMININ AKIS DIYAGRAMI



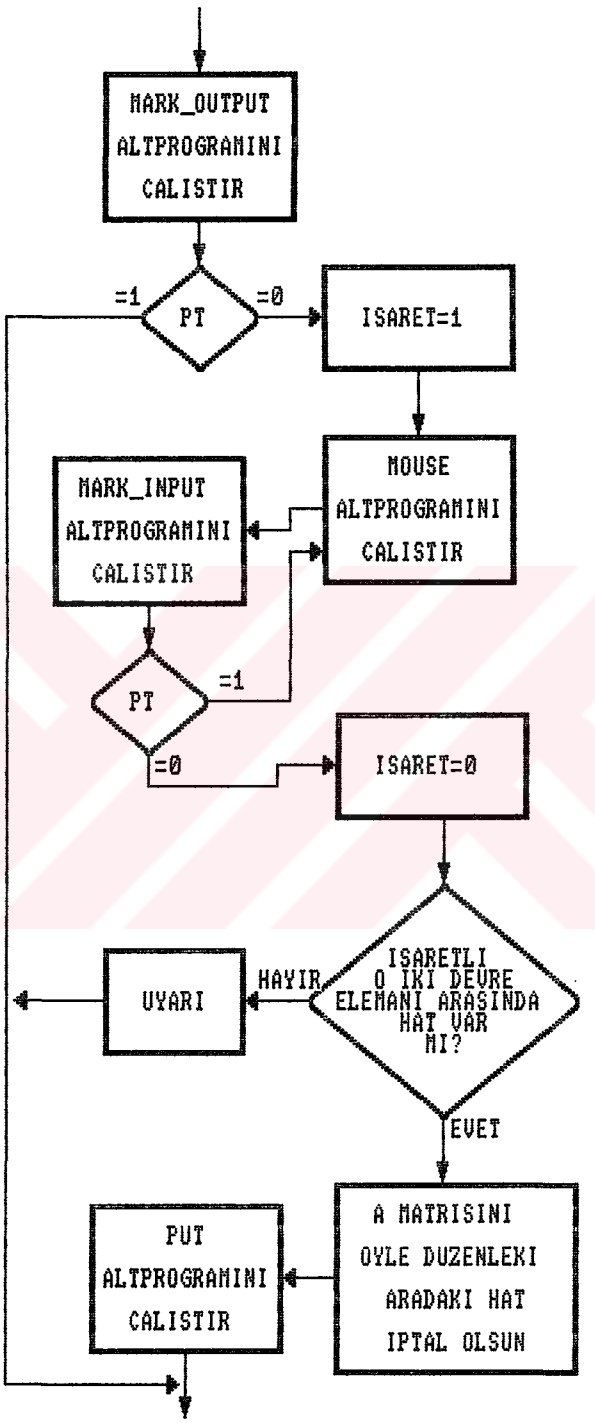
MARK_OUTPUT ALTPROGRAMININ AKIS DIYAGRAMI



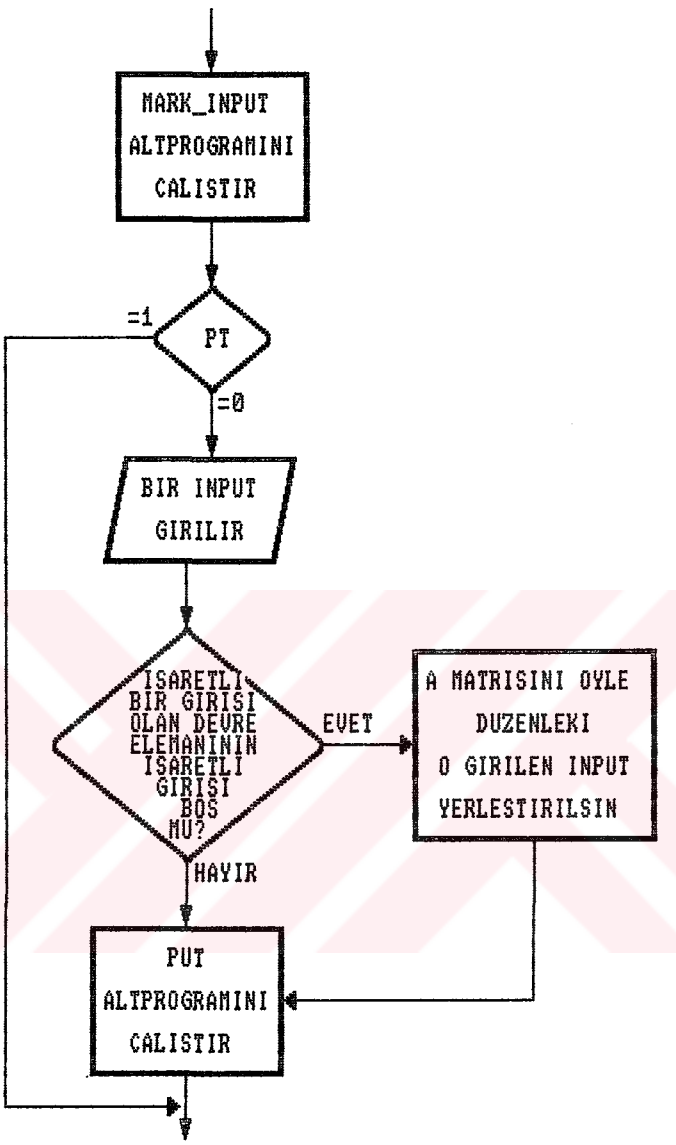
PUT_WIRE ALTPROGRAMININ AKIS DIYAGRAMI



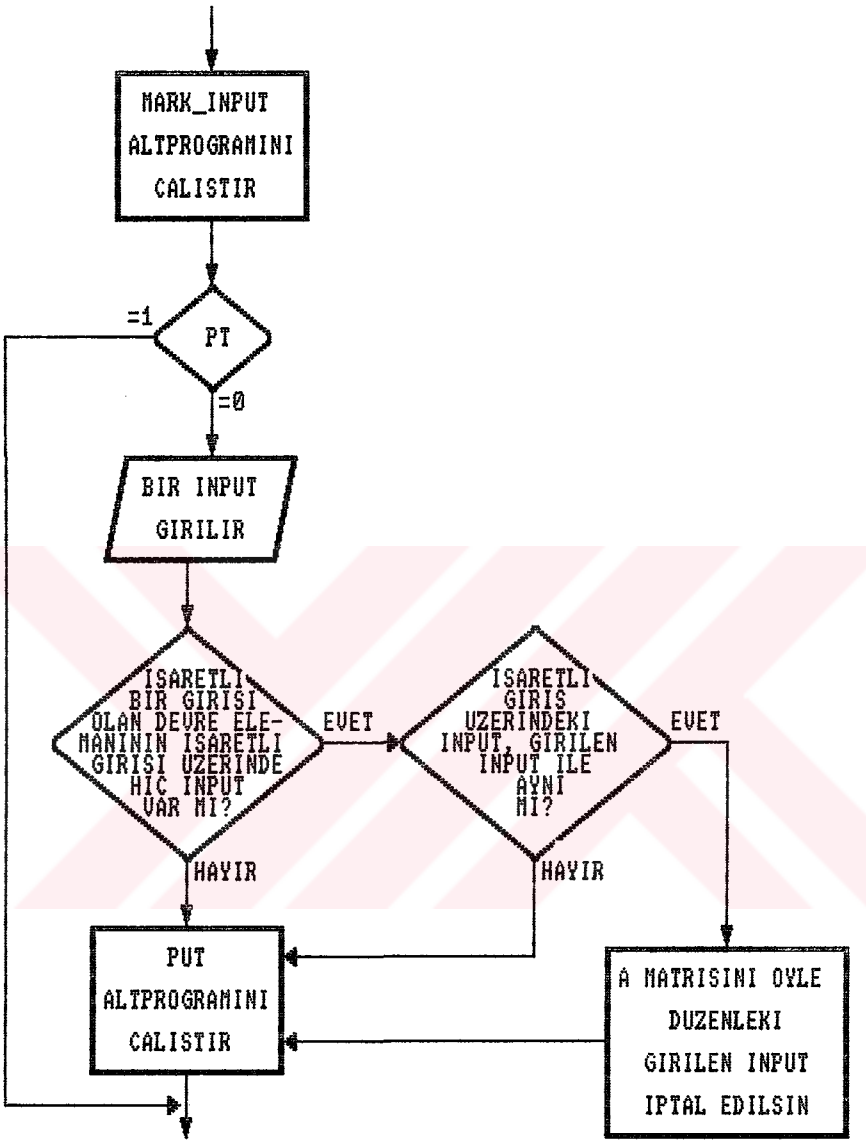
TAKE_WIRE ALTPROGRAMININ AKIS DIYAGRAMI



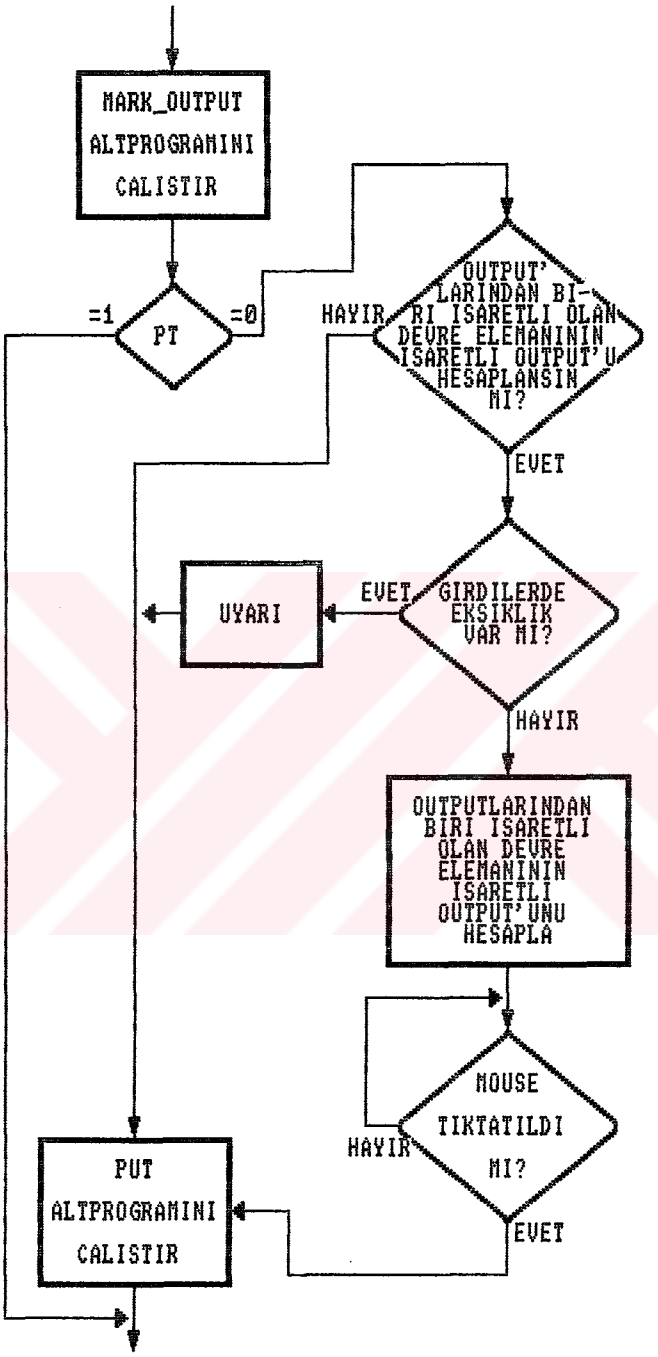
PUT_INPUT ALTPROGRAMININ AKIS DIYAGRAMI



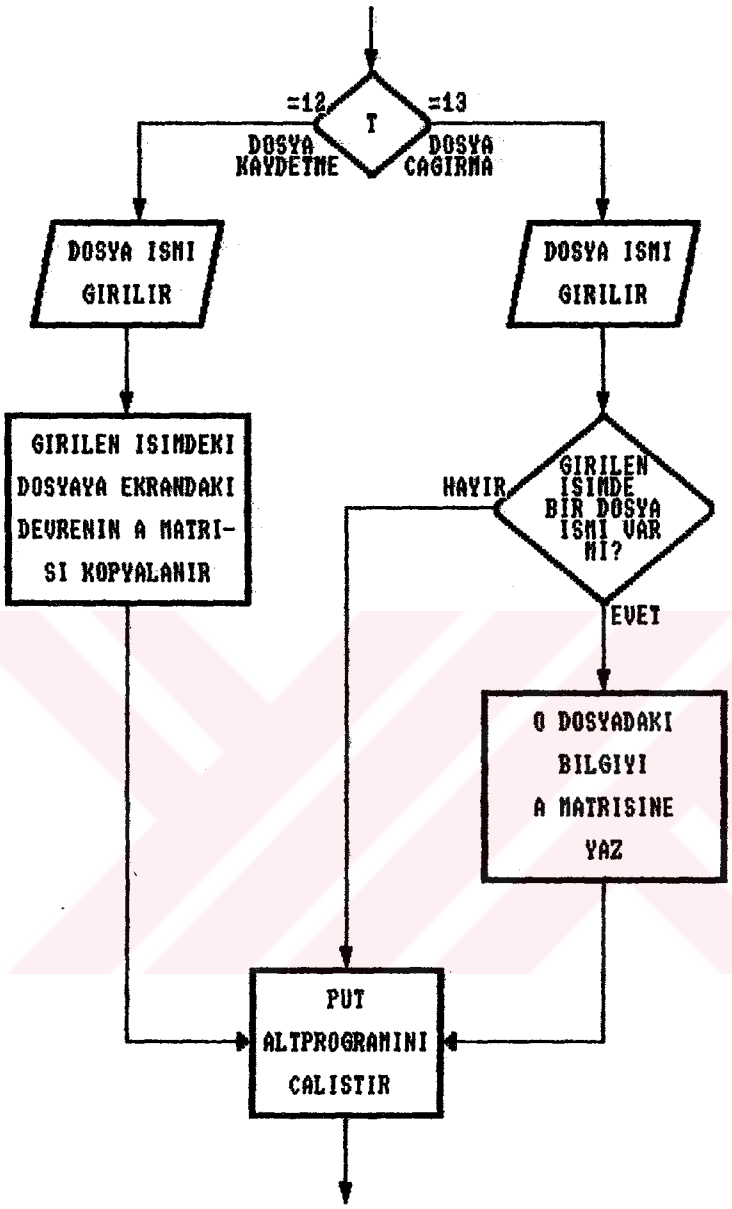
TAKE_INPUT ALTPROGRAMININ AKIS DIYAGRAMI



FIND_OUTPUT ALTPROGRAMININ AKIS DIYAGRAMI



SAVE_OR_TAKE ALTPROGRAMININ AKIS DIYAGRAMI



EK.2. PROGRAMIN LİSTESİ

1. LOGIC.PAS

```
USES CRT,GRAPH,DRIVERS,TEMURTAS;
CONST SON1=30;SON2=30;SON3=200;SON=3000;
TYPE MATRIX1=ARRAY[1..SON1,1..SON2] OF INTEGER;
     MATRIX2=ARRAY[1..SON] OF INTEGER;
     MATRIX3=ARRAY[1..SON1,1..SON3] OF INTEGER;
VAR DOSYA,DOSYA1,DOSYA2:TEXT;
    GD,GM,OKEY,IDOSYA,JDOSYA,YDOSYA,I,J,X,Y,T,TX,TY,YERX,YERY,
    CE,KUTULAR,YARDIM,SAY,SAYI,AX,AY,ISARET,GIRIS,CIKIS,PT,H,
    X1,X2,Y1,Y2,Y11,Y12,Y21,Y22,EK,HASANT,HT11,HT12,HT21,HT22,
    HT31,HT32,HT41,HT42,HT51,HT52,HT61,HT62,HTEMUR,TEMUR,G,G1,
    G2,G3,CIK,HELPX:INTEGER;
    A,O:MATRIX1;B:MATRIX3;Q1,Q2:MATRIX2;
    CH,CHDOS,CHDOSYA:CHAR;
    DOSYA_ISMI:STRING[12];DOSYA_ISMI1:STRING[8];
    DOSYA_ISMI2:STRING[4];MUSIC,CircuitElement:STRING;
PROCEDURE CIRCUIT_ELEMENT;
VAR SAYAC:INTEGER;
BEGIN
 HIDEMOUSE;SETCOLOR(7);SETFILLSTYLE(1,8);SETTEXTSTYLE(2,0,4);
 IF A[I,3]<>0 THEN
 BEGIN
  X:=A[I,1]-64*YERX;Y:=A[I,2]-96-64*YERY;
  CASE A[I,3] OF
   1:AND_KAPISI(X,Y);2:NAND_KAPISI(X,Y);3:OR_KAPISI(X,Y);
   4:NOR_KAPISI(X,Y);5:EXOR_KAPISI(X,Y);6:EXNOR_KAPISI(X,Y);
   7:NOT_KAPISI(X,Y);8:JKFF(X,Y);9:DFF(X,Y);10:TFF(X,Y);
   11:RSFF(X,Y);12:HALF_ADDER(X,Y);13:FULL_ADDER(X,Y);
   14:ENCODER21(X,Y);15:ENCODER42(X,Y);16:MULTIPLEXER21(X,Y);
   17:MULTIPLEXER41(X,Y);18:DEMULTIPLEXER12(X,Y);
   19:DEMULTIPLEXER14(X,Y);20:DECODER12(X,Y);21:DECODER24(X,Y);
   22:REGISTER2(X,Y);23:REGISTER4(X,Y);
  END;
  SAYI:=A[I,4]+A[I,13];
  IF (A[I,3]=1)OR(A[I,3]=2) THEN
```

```
    CASE SAYI OF
     1:LINE(X-6,Y,X,Y);
     2:BEGIN LINE(X-6,Y-8,X,Y-8);LINE(X-6,Y+8,X,Y+8) END;
     3:BEGIN
        LINE(X-6,Y-12,X,Y-12);LINE(X-6,Y,X,Y);LINE(X-6,Y+12,X,Y+12)
       END;
     4:BEGIN
        LINE(X-6,Y-15,X,Y-15);LINE(X-6,Y-5,X,Y-5);
        LINE(X-6,Y+5,X,Y+5);LINE(X-6,Y+15,X,Y+15)
       END;
    END;
   IF (A[I,3]=3)OR(A[I,3]=4)OR(A[I,3]=5)OR(A[I,3]=6) THEN
    CASE SAYI OF
     1:LINE(X-6,Y,X+6,Y);
     2:BEGIN LINE(X-6,Y-8,X+5,Y-8);LINE(X-6,Y+8,X+5,Y+8) END;
     3:BEGIN
        LINE(X-6,Y-12,X+5,Y-12);LINE(X-6,Y,X+6,Y);
        LINE(X-6,Y+12,X+5,Y+12)
       END;
     4:BEGIN
        LINE(X-6,Y-15,X+4,Y-15);LINE(X-6,Y-5,X+6,Y-5);
        LINE(X-6,Y+5,X+6,Y+5);LINE(X-6,Y+15,X+4,Y+15)
       END;
    END;
   IF ((A[I,3]>7)AND(A[I,3]<12))OR(A[I,3]=22)OR(A[I,3]=23) THEN
   BEGIN
    A[I,20]:=SAY+1;
    IF (A[I,3]>7)AND(A[I,3]<12) THEN SAYAC:=1;
    IF A[I,3]=22 THEN SAYAC:=2;IF A[I,3]=23 THEN SAYAC:=4;
    CEPART(X,Y,SAYAC,SAY);
   END;
 END;
 SHOWMOUSE;
END;
PROCEDURE GIRDI(GIR1,GIR2,GIR:INTEGER);
BEGIN
 CASE A[I,GIR] OF
```

```
  31:OUTTEXTXY(X+GIR1,Y+GIR2,'X');32:OUTTEXTXY(X+GIR1,Y+GIR2,'Y');
  33:OUTTEXTXY(X+GIR1,Y+GIR2,'Z');34:OUTTEXTXY(X+GIR1,Y+GIR2,'W');
  35:OUTTEXTXY(X+GIR1,Y+GIR2,'0');36:OUTTEXTXY(X+GIR1,Y+GIR2,'1');
END END;
PROCEDURE INPUTS;
BEGIN
 IF A[I,3]<>0 THEN
 BEGIN
  X:=A[I,1]-64*YERX;Y:=A[I,2]-64*YERY-96;
  CASE A[I,3] OF
   1..7:BEGIN
         SAYI:=A[I,4]+A[I,13];
         CASE SAYI OF
          1:IF A[I,13]=1 THEN GIRDI(-12,-6,14);
          2:BEGIN
             IF A[I,13]=1 THEN GIRDI(-12,2,14);
             IF A[I,13]=2 THEN
             BEGIN GIRDI(-12,-14,14);GIRDI(-12,2,15) END;
            END;
          3:BEGIN
             IF A[I,13]=1 THEN GIRDI(-12,6,14);
             IF A[I,13]=2 THEN
             BEGIN GIRDI(-12,-6,14);GIRDI(-12,6,15) END;
             IF A[I,13]=3 THEN
             BEGIN
              GIRDI(-12,-18,14);GIRDI(-12,-6,15);GIRDI(-12,6,16)
             END;
            END;
          4:BEGIN
             IF A[I,13]=1 THEN GIRDI(-12,9,14);
             IF A[I,13]=2 THEN
             BEGIN GIRDI(-12,-1,14);GIRDI(-12,9,15) END;
             IF A[I,13]=3 THEN
             BEGIN
              GIRDI(-12,-11,14);GIRDI(-12,-1,15);GIRDI(-12,9,16)
             END;
             IF A[I,13]=4 THEN
```

```
              BEGIN
               GIRDI(-12,-21,14);GIRDI(-12,-11,15);
               GIRDI(-12,-1,16);GIRDI(-12,+9,17);
         END END END END;
   8..12,14,18,22:BEGIN GIRDI(-12,-16,4);GIRDI(-12,4,6) END;
   20:GIRDI(-12,-6,4);
   21:BEGIN GIRDI(-19,-21,4);GIRDI(-19,9,6) END;
   13,16,19:BEGIN
       GIRDI(-19,-24,4);GIRDI(-19,-6,6);GIRDI(-19,12,8)
      END;
   15,23:BEGIN
          GIRDI(-19,-27,4);GIRDI(-19,-13,6);
          GIRDI(-19,1,8);GIRDI(-19,15,10);
         END;
   17:BEGIN
       GIRDI(-19,-26,4);GIRDI(-19,-18,6);GIRDI(-19,-10,8);
       GIRDI(-19,-2,10);GIRDI(-19,14,12);GIRDI(-19,22,14);
      END;
  END;
 END;
END;
PROCEDURE HAT(K:INTEGER);
VAR NOKTA:INTEGER;
BEGIN
 X2:=A[A[I,K],1]-64*YERX+44;Y2:=A[A[I,K],2]-64*YERY-96;
 CASE A[A[I,K],3] OF
  1..6,8..12,14,18,20,22:BEGIN Y21:=Y2-24;Y22:=Y2+24 END;
  7:BEGIN Y21:=Y2-16;Y22:=Y2+16 END;
  13,15,16,17,19,21,23:BEGIN Y21:=Y2-40;Y22:=Y2+40 END;
 END;
 CASE A[I,K+1] OF
  1:BEGIN X2:=X2+6;Y2:=Y2-10;LINE(X2-6,Y2,X2,Y2) END;
  2:BEGIN X2:=X2+9;Y2:=Y2+10;LINE(X2-9,Y2,X2,Y2) END;
  3:BEGIN X2:=X2+13;LINE(X2-6,Y2,X2,Y2) END;
  4:BEGIN X2:=X2+13;Y2:=Y2-15;LINE(X2-6,Y2,X2,Y2) END;
  5:BEGIN X2:=X2+16;Y2:=Y2+15;LINE(X2-9,Y2,X2,Y2) END;
  6:BEGIN X2:=X2+13;Y2:=Y2-21;LINE(X2-6,Y2,X2,Y2) END;
```

```
  7:BEGIN X2:=X2+16;Y2:=Y2-7;LINE(X2-9,Y2,X2,Y2) END;
  8:BEGIN X2:=X2+19;Y2:=Y2+7;LINE(X2-12,Y2,X2,Y2) END;
  9:BEGIN X2:=X2+22;Y2:=Y2+21;LINE(X2-15,Y2,X2,Y2) END;
 END;
 IF X1>X2 THEN
 BEGIN LINE(X2,Y2,X2,Y1);LINE(X2,Y1,X1,Y1) END ELSE
 BEGIN
  IF Y11>=Y22 THEN NOKTA:=Y22+EK ELSE IF Y11>=Y21 THEN NOKTA:=Y21-EK
  ELSE IF Y12>Y21 THEN
  BEGIN IF Y12<Y22 THEN NOKTA:=Y22+EK ELSE NOKTA:=Y12+EK END
  ELSE NOKTA:=Y21-EK;
  LINE(X2,Y2,X2,NOKTA);LINE(X2,NOKTA,X1,NOKTA);LINE(X1,NOKTA,X1,Y1);
 END;
END;
PROCEDURE LINES;
BEGIN
 IF A[I,3]<>0 THEN
 BEGIN
  SETCOLOR(7);X1:=A[I,1]-64*YERX-6;Y1:=A[I,2]-64*YERY-96;EK:=0;
  CASE A[I,3] OF
   1..6,8..12,14,18,20,22:BEGIN Y11:=Y1-24;Y12:=Y1+24 END;
   7:BEGIN Y11:=Y1-16;Y12:=Y1+16 END;
   13,15,16,17,19,21,23:BEGIN Y11:=Y1-40;Y12:=Y1+40 END;
  END;
  CASE A[I,3] OF
   1..7:BEGIN
    SAYI:=A[I,4]+A[I,13];
    CASE SAYI OF
     1:IF A[I,4]=1 THEN HAT(5);
     2:CASE A[I,4] OF
        1:BEGIN X1:=X1-8;Y1:=Y1-8;LINE(X1,Y1,X1+8,Y1);HAT(5) END;
        2:BEGIN
           X1:=X1-8;Y1:=Y1-8;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
           X1:=X1-3;Y1:=Y1+16;LINE(X1,Y1,X1+11,Y1);HAT(7)
          END;
       END;
     3:CASE A[I,4] OF
```

```
      1:BEGIN X1:=X1-8;Y1:=Y1-12;LINE(X1,Y1,X1+8,Y1);HAT(5) END;
      2:BEGIN
         X1:=X1-8;Y1:=Y1-12;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+12;LINE(X1,Y1,X1+11,Y1);HAT(7);
        END;
      3:BEGIN
         X1:=X1-8;Y1:=Y1-12;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+12;LINE(X1,Y1,X1+11,Y1);HAT(7);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+12;LINE(X1,Y1,X1+14,Y1);HAT(9);
        END;
     END;
  4:CASE A[I,4] OF
      1:BEGIN X1:=X1-8;Y1:=Y1-15;LINE(X1,Y1,X1+8,Y1);HAT(5) END;
      2:BEGIN
         X1:=X1-8;Y1:=Y1-15;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+11,Y1);HAT(7);
        END;
      3:BEGIN
         X1:=X1-8;Y1:=Y1-15;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+11,Y1);HAT(7);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+14,Y1);HAT(9);
        END;
      4:BEGIN
         X1:=X1-8;Y1:=Y1-15;LINE(X1,Y1,X1+8,Y1);HAT(5);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+11,Y1);HAT(7);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+14,Y1);HAT(9);EK:=EK+3;
         X1:=X1-3;Y1:=Y1+10;LINE(X1,Y1,X1+17,Y1);HAT(11);
        END;
     END;
       END;
 END;
20:IF (A[I,4]>0)AND(A[I,4]<31) THEN HAT(4);
9,10:IF (A[I,4]>0)AND(A[I,4]<31) THEN
 BEGIN X1:=X1-8;Y1:=Y1-10;LINE(X1,Y1,X1+8,Y1);HAT(4) END;
8,11,12,14,18,22:BEGIN
  X1:=X1-8;Y1:=Y1-10;
  IF (A[I,4]>0)AND(A[I,4]<31) THEN
```

```
   BEGIN LINE(X1,Y1,X1+8,Y1);HAT(4) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
   EK:=EK+3;X1:=X1-3;Y1:=Y1+20;
   IF (A[I,6]>0)AND(A[I,6]<31) THEN
   BEGIN LINE(X1,Y1,X1+11,Y1);HAT(6) END;
 END;
21:BEGIN
   X1:=X1-15;Y1:=Y1-15;
   IF (A[I,4]>0)AND(A[I,4]<31) THEN
   BEGIN LINE(X1,Y1,X1+8,Y1);HAT(4) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
   EK:=EK+3;X1:=X1-3;Y1:=Y1+30;
   IF (A[I,6]>0)AND(A[I,6]<31) THEN
   BEGIN LINE(X1,Y1,X1+11,Y1);HAT(6) END;
 END;
13,16,19:BEGIN
   X1:=X1-15;Y1:=Y1-18;
   IF (A[I,4]>0)AND(A[I,4]<31) THEN
   BEGIN LINE(X1,Y1,X1+8,Y1);HAT(4) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
   EK:=EK+3;X1:=X1-3;Y1:=Y1+18;
   IF (A[I,6]>0)AND(A[I,6]<31) THEN
   BEGIN LINE(X1,Y1,X1+11,Y1);HAT(6) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
   EK:=EK+3;X1:=X1-3;Y1:=Y1+18;
   IF (A[I,8]>0)AND(A[I,8]<31) THEN
   BEGIN LINE(X1,Y1,X1+14,Y1);HAT(8) END;
 END;
15,23:BEGIN
   X1:=X1-15;Y1:=Y1-21;
   IF (A[I,4]>0)AND(A[I,4]<31) THEN
   BEGIN LINE(X1,Y1,X1+8,Y1);HAT(4) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
   EK:=EK+3;X1:=X1-3;Y1:=Y1+14;
   IF (A[I,6]>0)AND(A[I,6]<31) THEN
   BEGIN LINE(X1,Y1,X1+11,Y1);HAT(6) END
   ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
```

```
      EK:=EK+3;X1:=X1-3;Y1:=Y1+14;
      IF (A[I,8]>0)AND(A[I,8]<31) THEN
      BEGIN LINE(X1,Y1,X1+14,Y1);HAT(8) END
      ELSE BEGIN X1:=X1+3;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-3;Y1:=Y1+14;
      IF (A[I,10]>0)AND(A[I,10]<31) THEN
      BEGIN LINE(X1,Y1,X1+17,Y1);HAT(10) END;
     END;
    17:BEGIN
      X1:=X1-15;Y1:=Y1-20;
      IF (A[I,4]>0)AND(A[I,4]<31) THEN
      BEGIN LINE(X1,Y1,X1+8,Y1);HAT(4) END
      ELSE BEGIN X1:=X1+2;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-2;Y1:=Y1+8;
      IF (A[I,6]>0)AND(A[I,6]<31) THEN
      BEGIN LINE(X1,Y1,X1+10,Y1);HAT(6) END
      ELSE BEGIN X1:=X1+2;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-2;Y1:=Y1+8;
      IF (A[I,8]>0)AND(A[I,8]<31) THEN
      BEGIN LINE(X1,Y1,X1+12,Y1);HAT(8) END
      ELSE BEGIN X1:=X1+2;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-2;Y1:=Y1+8;
      IF (A[I,10]>0)AND(A[I,10]<31) THEN
      BEGIN LINE(X1,Y1,X1+14,Y1);HAT(10) END
      ELSE BEGIN X1:=X1+2;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-2;Y1:=Y1+16;
      IF (A[I,12]>0)AND(A[I,12]<31) THEN
      BEGIN LINE(X1,Y1,X1+16,Y1);HAT(12) END
      ELSE BEGIN X1:=X1+2;EK:=EK-3 END;
      EK:=EK+3;X1:=X1-2;Y1:=Y1+8;
      IF (A[I,14]>0)AND(A[I,14]<31) THEN
      BEGIN LINE(X1,Y1,X1+18,Y1);HAT(14) END;
     END;
   END;
  END;
END;
```

```
PROCEDURE DUZENPART(DEGER1,DEGER2:INTEGER);
VAR DEG1,DEG2,DEGE1,DEGE2:INTEGER;
BEGIN
 DEG1:=0;DEG2:=0;
 CASE A[I,DEGER1+1] OF
  1:DEG1:=-10;2:DEG1:=+10;4:DEG1:=-15;5:DEG1:=+15;
  6:DEG1:=-21;7:DEG1:=-7;8:DEG1:=+7;9:DEG1:=+21;
 END;
 CASE A[I,DEGER2+1] OF
  1:DEG2:=-10;2:DEG2:=+10;4:DEG2:=-15;5:DEG2:=+15;
  6:DEG2:=-21;7:DEG2:=-7;8:DEG2:=+7;9:DEG2:=+21;
 END;
 IF A[A[I,DEGER1],2]+DEG1>A[A[I,DEGER2],2]+DEG2 THEN
 BEGIN
DEGE1:=A[I,DEGER1];DEGE2:=A[I,DEGER1+1];A[I,DEGER1]:=A[I,DEGER2];
A[I,DEGER1+1]:=A[I,DEGER2+1];A[I,DEGER2]:=DEGE1;A[I,DEGER2+1]:=DEGE2;
 END;
END;
PROCEDURE DUZEN;
BEGIN
 FOR I:=1 TO SON1 DO IF (A[I,3]>0)AND(A[I,3]<7) THEN
 CASE A[I,4] OF
  2:DUZENPART(5,7);
  3:BEGIN DUZENPART(5,7);DUZENPART(7,9);DUZENPART(5,7) END;
  4:BEGIN
     DUZENPART(5,7);DUZENPART(7,9);DUZENPART(9,11);
     DUZENPART(5,7);DUZENPART(7,9);DUZENPART(5,7);
    END;
 END;
END;
PROCEDURE PUT;
BEGIN
 DUZEN;HIDEMOUSE;SETVIEWPORT(0,96,639,479,TRUE);CLEARVIEWPORT;SAY:=0;
 FOR I:=1 TO SON1 DO CIRCUIT_ELEMENT;
 FOR I:=1 TO SON1 DO INPUTS;FOR I:=1 TO SON1 DO LINES;
 SETVIEWPORT(0,0,639,479,TRUE);SETCOLOR(15);
 IF YERX=0 THEN BEGIN LINE(0,96,0,479);LINE(1,96,1,479) END;
```

```
 IF YERX=4 THEN BEGIN LINE(638,96,638,479);LINE(639,96,639,479) END;
 IF YERY=0 THEN BEGIN LINE(0,96,639,96);LINE(0,97,639,97) END;
 IF YERY=4 THEN BEGIN LINE(0,478,639,478);LINE(0,479,639,479) END;
 SHOWMOUSE;
END;
PROCEDURE MOUSE;
LABEL 1,2;
VAR RCE,MCE,SAKLA,RT,MT,KAYDIR,SES,YER1,YER2:INTEGER;
BEGIN
 IF ISARET=0 THEN
 BEGIN
  TX:=T;
  1:REPEAT
   X:=8*MOUSEWHERE.X;Y:=8*MOUSEWHERE.Y;CH:=' ';
   CASE KUTULAR OF
    0:BEGIN
       KAYDIR:=0;SES:=0;
       IF KEYPRESSED THEN
       BEGIN
        CH:=READKEY;IF CH=#0 THEN CH:=READKEY;
       END ELSE
       IF Y<24 THEN
       BEGIN
        IF X<160 THEN RT:=1 ELSE
        IF X<200 THEN BEGIN RT:=14;KAYDIR:=1 END ELSE RT:=0
       END ELSE IF Y<48 THEN
       BEGIN
        IF X<40 THEN RT:=2 ELSE IF X<80 THEN RT:=5 ELSE
        IF X<120 THEN RT:=8 ELSE IF X<160 THEN BEGIN RT:=11;SES:=1 END
        ELSE IF X<200 THEN BEGIN RT:=15;KAYDIR:=2 END ELSE RT:=0
       END ELSE IF Y<72 THEN
       BEGIN
        IF X<40 THEN RT:=3 ELSE IF X<80 THEN RT:=6 ELSE
        IF X<120 THEN RT:=9 ELSE IF X<160 THEN RT:=12 ELSE
        IF X<200 THEN BEGIN RT:=16;KAYDIR:=3 END ELSE RT:=0
       END ELSE IF Y<96 THEN
       BEGIN
```

```
    IF X<40 THEN RT:=4 ELSE IF X<80 THEN RT:=7 ELSE
    IF X<120 THEN RT:=10 ELSE IF X<160 THEN RT:=13 ELSE
    IF X<200 THEN BEGIN RT:=17;KAYDIR:=4 END ELSE RT:=0
   END ELSE RT:=0;
   IF (RT<>MT)OR((RT=0)AND(MT<>0)) THEN
   BEGIN
    IF MUSIC='SOUND' THEN
    BEGIN
     IF (RT<>0)OR(MT<>0) THEN SOUND(1000) ELSE SOUND(2000);
     DELAY(50);NOSOUND
    END;
    SAKLA:=T;T:=RT;
    IF T<>MT THEN
    BEGIN
     RENK0(T,14,CircuitElement);T:=MT;RENK0(T,5,CircuitElement);
     T:=SAKLA;RENK0(T,11,CircuitElement)
    END;
    T:=SAKLA;MT:=RT;
   END;
  END;
1:BEGIN
   IF Y<24 THEN
   BEGIN
    IF X<80 THEN RCE:=1 ELSE IF X<160 THEN RCE:=2 ELSE
    IF X<240 THEN RCE:=3 ELSE IF X<320 THEN RCE:=4 ELSE
    IF X<400 THEN RCE:=5 ELSE IF X<480 THEN RCE:=6 ELSE
    IF X<560 THEN RCE:=7 ELSE IF X<640 THEN RCE:=8
   END ELSE IF Y<48 THEN
   BEGIN
    IF X<80 THEN RCE:=9 ELSE IF X<160 THEN RCE:=10 ELSE
    IF X<240 THEN RCE:=11 ELSE IF X<320 THEN RCE:=12 ELSE
    IF X<400 THEN RCE:=13 ELSE IF X<480 THEN RCE:=14 ELSE
    IF X<560 THEN RCE:=15 ELSE IF X<640 THEN RCE:=16
   END ELSE IF Y<72 THEN
   BEGIN
    IF X<80 THEN RCE:=17 ELSE IF X<160 THEN RCE:=18 ELSE
    IF X<240 THEN RCE:=19 ELSE IF X<320 THEN RCE:=20 ELSE
```

```
      IF X<400 THEN RCE:=21 ELSE IF X<480 THEN RCE:=22 ELSE
      IF X<560 THEN RCE:=23 ELSE IF X<640 THEN RCE:=24
     END ELSE IF Y<96 THEN
     BEGIN
      IF X<80 THEN RCE:=25 ELSE IF X<160 THEN RCE:=26 ELSE
      IF X<240 THEN RCE:=27 ELSE IF X<320 THEN RCE:=28 ELSE
      IF X<400 THEN RCE:=29 ELSE IF X<480 THEN RCE:=30 ELSE
      IF X<560 THEN RCE:=31 ELSE IF X<640 THEN RCE:=32
     END ELSE RCE:=0;
     IF (RCE<>MCE)OR((RCE=0)AND(MCE<>0)) THEN
     BEGIN
      IF MUSIC='SOUND' THEN
      BEGIN
       IF (RCE<>0)OR(MCE<>0) THEN SOUND(1000) ELSE SOUND(2000);
       DELAY(50);NOSOUND
      END;
      SAKLA:=CE;CE:=RCE;
      IF CE<>MCE THEN
      BEGIN
       RENK1(CE,14);CE:=MCE;RENK1(CE,5);CE:=SAKLA;RENK1(CE,11)
      END;
      CE:=SAKLA;MCE:=RCE;
     END;
     IF (RCE=0)OR((RCE>23)AND(RCE<33)) THEN GOTO 1;
    END
 END;
UNTIL (MOUSEBUTTONS<>EVNOTHING)OR(CH=#27)OR(CH=#56)OR(CH=#50)OR
      (CH=#54)OR(CH=#52)OR(CH=#72)OR(CH=#80)OR(CH=#77)OR(CH=#75);
CASE KUTULAR OF
 0:BEGIN
    IF CH=#27 THEN BEGIN X:=72;Y:=88;RT:=7 END;
    IF (CH=#56)OR(CH=#72) THEN
    BEGIN X:=192;Y:=16;RT:=14;KAYDIR:=1 END;
    IF (CH=#50)OR(CH=#80) THEN
    BEGIN X:=192;Y:=40;RT:=15;KAYDIR:=2 END;
    IF (CH=#54)OR(CH=#77) THEN
    BEGIN X:=192;Y:=64;RT:=16;KAYDIR:=3 END;
```

```
 IF (CH=#52)OR(CH=#75) THEN
 BEGIN X:=192;Y:=88;RT:=17;KAYDIR:=4 END;
 RENK0(T,5,CircuitElement);IF RT<>0 THEN T:=RT;
 IF (T=1)AND(RT<>0) THEN
BEGIN KUTULAR:=1;BOXES1;RENK1(CE,11);DELAY(250);TX:=T;GOTO 1 END;
 IF (Y<96)AND(X>199) THEN
 BEGIN RENK0(T,11,CircuitElement);GOTO 1 END;
 RENK0(T,11,CircuitElement);
 IF MUSIC='SOUND' THEN
 BEGIN
  SOUND(80);DELAY(25);SOUND(50);DELAY(25);SOUND(40);
  DELAY(25);SOUND(30);DELAY(25);NOSOUND;DELAY(150)
 END ELSE DELAY(250);
 IF SES<>0 THEN
 BEGIN
  IF MUSIC='SOUND' THEN MUSIC:='NOSOUND' ELSE MUSIC:='SOUND';
  DELAY(250);RENK0(T,5,CircuitElement);T:=TX;
  RENK0(T,11,CircuitElement);GOTO 1;
 END;
 IF KAYDIR<>0 THEN
 BEGIN
  HIDEMOUSE;YER1:=YERX;YER2:=YERY;
  CASE KAYDIR OF
   1:IF YERY>0 THEN YERY:=YERY-1;
   2:IF YERY<4 THEN YERY:=YERY+1;
   3:IF YERX<4 THEN YERX:=YERX+1;
   4:IF YERX>0 THEN YERX:=YERX-1;
  END;
  IF NOT((YERX=YER1)AND(YERY=YER2)) THEN
  BEGIN KONUM(YERX,YERY);PUT END;
  RENK0(T,5,CircuitElement);T:=TX;RENK0(T,11,CircuitElement);
  SHOWMOUSE;GOTO 1;
 END;
 CASE T OF
  1:YARDIM:=1;2:YARDIM:=3;3:YARDIM:=4;4:YARDIM:=5;5:YARDIM:=7;
  8:YARDIM:=9;9:YARDIM:=11;10:YARDIM:=15;
 END;
```

```
      IF T<>TX THEN
      BEGIN
       HELP(YARDIM,HELPX);
       IF T<>1 THEN
       BEGIN
        HIDEMOUSE;SETVIEWPORT(3,3,155,19,TRUE);CLEARVIEWPORT;
        SETVIEWPORT(0,0,639,479,TRUE);SETFILLSTYLE(1,7);
        SETCOLOR(8);FLOODFILL(5,5,8);SETCOLOR(5);SETTEXTSTYLE(2,0,4);
        CircuitElement:='Put Circuit Element';
        OUTTEXTXY(25,5,CircuitElement);SHOWMOUSE
       END;
      END;
     END;
   1:BEGIN
      CE:=RCE;MOUSEPART(CE,CircuitElement);YARDIM:=1;KUTULAR:=0;
      BOXES0(CircuitElement);RENK0(T,11,CircuitElement);
      KONUM(YERX,YERY);HELP(YARDIM,HELPX);DELAY(250);GOTO 1;
     END
  END;
  X:=8*MOUSEWHERE.X;Y:=8*MOUSEWHERE.Y;
 END ELSE
 BEGIN
  REPEAT
   DELAY(100);
   2:X:=8*MOUSEWHERE.X;Y:=8*MOUSEWHERE.Y;
   IF Y<96 THEN GOTO 2;
  UNTIL MOUSEBUTTONS<>EVNOTHING;
END END;
PROCEDURE NEW;
BEGIN
 HIDEMOUSE;NEWPART;
 WHILE KEYPRESSED DO CH:=UPCASE(READKEY);CH:=' ';
 REPEAT
  SETCOLOR(0);OUTTEXTXY(525,10,CH);
  IF KEYPRESSED THEN CH:=UPCASE(READKEY);
  SETCOLOR(15);OUTTEXTXY(525,10,CH);DELAY(200);
 UNTIL (CH='Y')OR(CH='N');
```

```
 IF CH='Y' THEN
 BEGIN FOR I:=1 TO SON1 DO FOR J:=1 TO SON2 DO A[I,J]:=0;PUT END;
 RENK0(T,5,CircuitElement);T:=TX;RENK0(T,11,CircuitElement);
 HELP(YARDIM,HELPX);PARCA(CE,T,CircuitElement);SHOWMOUSE
END;
PROCEDURE GOOUT;
BEGIN
 HIDEMOUSE;GOOUTPART;
 WHILE KEYPRESSED DO CH:=UPCASE(READKEY);CH:=' ';
 REPEAT
  SETCOLOR(0);OUTTEXTXY(608,10,CH);
  IF KEYPRESSED THEN CH:=UPCASE(READKEY);
  SETCOLOR(15);OUTTEXTXY(608,10,CH);DELAY(200);
 UNTIL (CH='Y')OR(CH='N');
 RENK0(T,5,CircuitElement);
 IF CH='Y' THEN T:=-1 ELSE BEGIN T:=TX;RENK0(T,11,CircuitElement) END;
 HELP(YARDIM,HELPX);PARCA(CE,T,CircuitElement);SHOWMOUSE;
END;
PROCEDURE PUT_CIRCUIT_ELEMENT;
VAR PCE,PCE1,PCE2:INTEGER;
BEGIN
 FOR I:=1 TO SON1 DO IF A[I,3]<>0 THEN
 BEGIN
  IF CE=7 THEN PCE1:=1;IF CE<7 THEN PCE1:=2;
  IF ((CE>7)AND(CE<13))OR(CE=14)OR(CE=18)OR(CE=20)
     OR(CE=22) THEN PCE1:=3;
  IF (CE=15)OR(CE=17)OR(CE=19)OR(CE=21)OR(CE=23)OR
     (CE=13)OR(CE=16)  THEN PCE1:=4;
  IF A[I,3]=7 THEN PCE2:=10;
  IF A[I,3]<7 THEN PCE2:=20;
  IF ((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)OR(A[I,3]=18)
     OR(A[I,3]=20)OR(A[I,3]=22) THEN PCE2:=30;
  IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)
     OR(A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN PCE2:=40;
  PCE:=PCE1+PCE2;PCE1:=80;
  CASE PCE OF
   11:BEGIN PCE1:=72;PCE2:=40 END;
```

```
    12,21:BEGIN PCE1:=72;PCE2:=48 END;
    13,31:PCE2:=48;
    22,23,32,33:PCE2:=56;
    14,41:BEGIN PCE1:=88;PCE2:=64 END;
    24,42,34,43:BEGIN PCE1:=88;PCE2:=72 END;
    44:BEGIN PCE1:=96;PCE2:=88 END;
   END;
   IF (X>A[I,1]-64*YERX-PCE1)AND(X<A[I,1]-64*YERX+PCE1)AND
      (Y>A[I,2]-64*YERY-PCE2)AND(Y<A[I,2]-64*YERY+PCE2) THEN EXIT;
  END;
  IF Y>=96 THEN
  BEGIN
   I:=0;REPEAT I:=I+1 UNTIL (I=SON1+1)OR(A[I,3]=0);
   IF I=SON1+1 THEN BEGIN YARDIM:=2;HELP(YARDIM,HELPX);EXIT END;
   IF (YERX=0)AND(X<25) THEN EXIT;
   IF (YERX=4)AND(X>570) THEN EXIT;
   IF (YERY=0)AND(Y<150) THEN EXIT;
   IF (YERY=4)AND(Y>430) THEN EXIT;
   A[I,1]:=X+64*YERX;A[I,2]:=Y+64*YERY;A[I,3]:=CE;
   SETVIEWPORT(0,96,639,479,TRUE);CIRCUIT_ELEMENT;
   SETVIEWPORT(0,0,639,479,TRUE);
  END;
END;
PROCEDURE TAKE_CIRCUIT_ELEMENT;
LABEL 1;VAR TCE1,TCE2,HTEMUR1,HTEMUR2:INTEGER;
BEGIN
 FOR I:=1 TO SON1 DO IF A[I,3]<>0 THEN
 BEGIN
  IF A[I,3]=7 THEN
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+8) THEN GOTO 1;
  IF (A[I,3]<7)OR((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)OR(A[I,3]=18)
     OR(A[I,3]=20)OR(A[I,3]=22) THEN
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)
     OR(A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN
```

```
  IF (X>A[I,1]-64*YERX-16)AND(X<A[I,1]-64*YERX+40)AND
    (Y>A[I,2]-64*YERY-40)AND(Y<A[I,2]-64*YERY+32) THEN GOTO 1;
END;
EXIT;
1:FOR TCE1:=1 TO SON1 DO IF A[TCE1,3]<>0 THEN IF A[TCE1,3]<8 THEN
BEGIN
 IF A[TCE1,5]=I THEN
 BEGIN A[TCE1,4]:=A[TCE1,4]-1;A[TCE1,5]:=0;A[TCE1,6]:=0 END;
 IF A[TCE1,7]=I THEN
 BEGIN A[TCE1,4]:=A[TCE1,4]-1;A[TCE1,7]:=0;A[TCE1,8]:=0 END;
 IF A[TCE1,9]=I THEN
 BEGIN A[TCE1,4]:=A[TCE1,4]-1;A[TCE1,9]:=0;A[TCE1,10]:=0 END;
 IF A[TCE1,11]=I THEN
 BEGIN A[TCE1,4]:=A[TCE1,4]-1;A[TCE1,11]:=0;A[TCE1,12]:=0 END;
 IF A[TCE1,9]=0 THEN
 BEGIN
  A[TCE1,9]:=A[TCE1,11];A[TCE1,10]:=A[TCE1,12];A[TCE1,11]:=0;
  A[TCE1,12]:=0
 END;
 IF A[TCE1,7]=0 THEN
 BEGIN
  A[TCE1,7]:=A[TCE1,9];A[TCE1,8]:=A[TCE1,10];A[TCE1,9]:=A[TCE1,11];
  A[TCE1,10]:=A[TCE1,12];A[TCE1,11]:=0;A[TCE1,12]:=0
 END;
 IF A[TCE1,5]=0 THEN
 BEGIN
  A[TCE1,5]:=A[TCE1,7];A[TCE1,6]:=A[TCE1,8];A[TCE1,7]:=A[TCE1,9];
  A[TCE1,8]:=A[TCE1,10];A[TCE1,9]:=A[TCE1,11];A[TCE1,10]:=A[TCE1,12];
  A[TCE1,11]:=0;A[TCE1,12]:=0;
 END;
END ELSE
BEGIN
 IF A[TCE1,4]=I THEN BEGIN A[TCE1,4]:=0;A[TCE1,5]:=0 END;
 IF A[TCE1,6]=I THEN BEGIN A[TCE1,6]:=0;A[TCE1,7]:=0 END;
 IF A[TCE1,8]=I THEN BEGIN A[TCE1,8]:=0;A[TCE1,9]:=0 END;
 IF A[TCE1,10]=I THEN BEGIN A[TCE1,10]:=0;A[TCE1,11]:=0 END;
 IF A[TCE1,12]=I THEN BEGIN A[TCE1,12]:=0;A[TCE1,13]:=0 END;
```

```
  IF A[TCE1,14]=I THEN BEGIN A[TCE1,14]:=0;A[TCE1,15]:=0 END;
 END;
 FOR TCE2:=1 TO SON2 DO A[I,TCE2]:=0;PUT;
END;
PROCEDURE MOVE_CIRCUIT_ELEMENT;
LABEL 1,2;
VAR MCE,MCE1,MCE2,MCE3:INTEGER;
BEGIN
 FOR I:=1 TO SON1 DO IF A[I,3]<>0 THEN
 BEGIN
  IF A[I,3]=7 THEN
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+8) THEN GOTO 1;
  IF (A[I,3]<7)OR((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)
     OR(A[I,3]=18)OR(A[I,3]=20)OR(A[I,3]=22) THEN
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)OR
     (A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN
   IF (X>A[I,1]-64*YERX-16)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-40)AND(Y<A[I,2]-64*YERY+32) THEN GOTO 1;
 END;
 EXIT;
 1:ISARET:=1;MCE:=I;AX:=A[I,1]-64*YERX;
 AY:=A[I,2]-64*YERY;MARK(AX,AY,1);
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 2:MOUSE;
 FOR I:=1 TO SON1 DO
 BEGIN
  IF I=MCE THEN I:=I+1;
  IF (A[I,3]<>0)AND(I<31) THEN
  BEGIN
   IF CE=7 THEN MCE1:=1;IF CE<7 THEN MCE1:=2;
   IF ((CE>7)AND(CE<13))OR(CE=14)OR(CE=18)OR(CE=20)OR(CE=22)
   THEN MCE1:=3;
   IF (CE=15)OR(CE=17)OR(CE=19)OR(CE=21)OR(CE=23)OR
      (CE=13)OR(CE=16) THEN MCE1:=4;
```

```
    IF A[I,3]=7 THEN MCE2:=10;
    IF A[I,3]<7 THEN MCE2:=20;
    IF ((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)OR(A[I,3]=18)
       OR(A[I,3]=20)OR(A[I,3]=22) THEN MCE2:=30;
    IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)
       OR(A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN MCE2:=40;
    MCE3:=MCE1+MCE2;MCE1:=80;
    CASE MCE3 OF
     11:BEGIN MCE1:=72;MCE2:=40 END;
     12,21:BEGIN MCE1:=72;MCE2:=48 END;
     13,31:MCE2:=48;
     22,23,32,33:MCE2:=56;
     14,41:BEGIN MCE1:=88;MCE2:=64 END;
     24,42,34,43:BEGIN MCE1:=88;MCE2:=72 END;
     44:BEGIN MCE1:=96;MCE2:=88 END;
    END;
    IF (X>A[I,1]-64*YERX-MCE1)AND(X<A[I,1]-64*YERX+MCE1)AND
      (Y>A[I,2]-64*YERY-MCE2)AND(Y<A[I,2]-64*YERY+MCE2) THEN GOTO 2;
  END;
 END;
 IF (YERX=0)AND(X<25) THEN GOTO 2;
 IF (YERX=4)AND(X>570) THEN GOTO 2;
 IF (YERY=0)AND(Y<150) THEN GOTO 2;
 IF (YERY=4)AND(Y>430) THEN GOTO 2;
 ISARET:=0;A[MCE,1]:=X+64*YERX;A[MCE,2]:=Y+64*YERY;
 PUT;REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
END;
PROCEDURE MARK_INPUT;
LABEL 1;
BEGIN
 PT:=0;
 FOR I:=1 TO SON1 DO IF A[I,3]<>0 THEN
 BEGIN
  IF A[I,3]=7 THEN
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+24)AND
     (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+8) THEN GOTO 1;
  IF A[I,3]<7 THEN
```

```
   IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+32)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF ((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)OR(A[I,3]=18)
     OR(A[I,3]=20)OR(A[I,3]=22) THEN
  IF (X>A[I,1]-64*YERX-8)AND(X<A[I,1]-64*YERX+16)AND
     (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)
     OR(A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN
    IF (X>A[I,1]-64*YERX-16)AND(X<A[I,1]-64*YERX+16)AND
     (Y>A[I,2]-64*YERY-40)AND(Y<A[I,2]-64*YERY+32) THEN GOTO 1;
END;
PT:=1;EXIT;
1:AX:=A[I,1]-64*YERX;AY:=A[I,2]-64*YERY;
IF A[I,3]<8 THEN BEGIN GIRIS:=0;MARK(AX,AY,1) END;
IF A[I,3]=20 THEN BEGIN GIRIS:=1;MARK(AX,AY,1) END;
IF (A[I,3]=9)OR(A[I,3]=10) THEN BEGIN GIRIS:=1;MARK(AX,AY,2) END;
IF (A[I,3]=8)OR(A[I,3]=11)OR(A[I,3]=12)OR(A[I,3]=14)OR
   (A[I,3]=18)OR(A[I,3]=22) THEN
 IF Y<A[I,2]-64*YERY THEN BEGIN GIRIS:=1;MARK(AX,AY,2) END
 ELSE BEGIN GIRIS:=2;MARK(AX,AY,4) END;
IF A[I,3]=21 THEN
 IF Y<A[I,2]-64*YERY THEN BEGIN GIRIS:=1;MARK(AX,AY,18) END
 ELSE BEGIN GIRIS:=2;MARK(AX,AY,19) END;
IF (A[I,3]=13)OR(A[I,3]=16)OR(A[I,3]=19) THEN
 IF Y<A[I,2]-64*YERY-8 THEN BEGIN GIRIS:=1;MARK(AX,AY,15) END
 ELSE IF Y<A[I,2]-64*YERY+8 THEN BEGIN GIRIS:=2;MARK(AX,AY,16) END
 ELSE BEGIN GIRIS:=3;MARK(AX,AY,17) END;
IF (A[I,3]=15)OR(A[I,3]=23) THEN
 IF Y<A[I,2]-64*YERY-16 THEN BEGIN GIRIS:=1;MARK(AX,AY,11) END
 ELSE IF Y<A[I,2]-64*YERY THEN BEGIN GIRIS:=2;MARK(AX,AY,12) END
 ELSE IF Y<A[I,2]-64*YERY+16 THEN BEGIN GIRIS:=3;MARK(AX,AY,13) END
 ELSE BEGIN GIRIS:=4;MARK(AX,AY,14) END;
IF A[I,3]=17 THEN
 IF Y<A[I,2]-64*YERY-16 THEN BEGIN GIRIS:=1;MARK(AX,AY,5) END
 ELSE IF Y<A[I,2]-64*YERY-8 THEN BEGIN GIRIS:=2;MARK(AX,AY,6) END
 ELSE IF Y<A[I,2]-64*YERY THEN BEGIN GIRIS:=3;MARK(AX,AY,7) END
 ELSE IF Y<A[I,2]-64*YERY+8 THEN BEGIN GIRIS:=4;MARK(AX,AY,8) END
```

```
  ELSE IF Y<A[I,2]-64*YERY+24 THEN BEGIN GIRIS:=5;MARK(AX,AY,9) END
  ELSE BEGIN GIRIS:=6;MARK(AX,AY,10) END;
END;
PROCEDURE MARK_OUTPUT;
LABEL 1;
BEGIN
 PT:=0;
 FOR I:=1 TO SON1 DO IF A[I,3]<>0 THEN
 BEGIN
  IF A[I,3]=7 THEN IF (X>A[I,1]-64*YERX)AND(X<A[I,1]-64*YERX+40)
  AND(Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+8) THEN GOTO 1;
  IF A[I,3]<7 THEN
   IF (X>A[I,1]-64*YERX)AND(X<A[I,1]-64*YERX+40)AND
     (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF ((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=14)
     OR(A[I,3]=18)OR(A[I,3]=20)OR(A[I,3]=22) THEN
   IF (X>A[I,1]-64*YERX+16)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-24)AND(Y<A[I,2]-64*YERY+16) THEN GOTO 1;
  IF (A[I,3]=15)OR(A[I,3]=17)OR(A[I,3]=19)OR(A[I,3]=21)OR
     (A[I,3]=23)OR(A[I,3]=13)OR(A[I,3]=16) THEN
   IF (X>A[I,1]-64*YERX+16)AND(X<A[I,1]-64*YERX+40)AND
      (Y>A[I,2]-64*YERY-40)AND(Y<A[I,2]-64*YERY+32) THEN GOTO 1;
 END;
 PT:=1;EXIT;1:AX:=A[I,1]-64*YERX;AY:=A[I,2]-64*YERY;
 IF A[I,3]<8 THEN BEGIN CIKIS:=0;MARK(AX,AY,30) END;
 IF ((A[I,3]>7)AND(A[I,3]<13))OR(A[I,3]=18)OR(A[I,3]=20)OR
    (A[I,3]=22) THEN
  IF Y<A[I,2]-64*YERY THEN BEGIN CIKIS:=1;MARK(AX,AY,20) END
  ELSE BEGIN CIKIS:=2;MARK(AX,AY,22) END;
 IF (A[I,3]=13)OR(A[I,3]=15) THEN
  IF Y<A[I,2]-64*YERY THEN BEGIN CIKIS:=4;MARK(AX,AY,27) END
  ELSE BEGIN CIKIS:=5;MARK(AX,AY,28) END;
 IF A[I,3]=14 THEN BEGIN CIKIS:=0;MARK(AX,AY,21) END;
 IF (A[I,3]=16)OR (A[I,3]=17) THEN BEGIN CIKIS:=3;MARK(AX,AY,29) END;
 IF (A[I,3]=19)OR(A[I,3]=21)OR(A[I,3]=23) THEN
  IF Y<A[I,2]-64*YERY-16 THEN BEGIN CIKIS:=6;MARK(AX,AY,23) END
  ELSE IF Y<A[I,2]-64*YERY THEN BEGIN CIKIS:=7;MARK(AX,AY,24) END
```

```
  ELSE IF Y<A[I,2]-64*YERY+16 THEN BEGIN CIKIS:=8;MARK(AX,AY,25) END
  ELSE BEGIN CIKIS:=9;MARK(AX,AY,26) END;
END;
PROCEDURE PUT_WIRE;
LABEL 1,2,3,4;
VAR PW,I1,I2,I3,I4:INTEGER;
    C:ARRAY[1..180,1..2] OF INTEGER;
BEGIN
 MARK_OUTPUT;IF PT=1 THEN EXIT;ISARET:=1;PW:=I;
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 REPEAT MOUSE;MARK_INPUT UNTIL PT=0;
 ISARET:=0;
 CASE A[I,3] OF
  1..7:BEGIN
        SAYI:=A[I,4]+A[I,13];
        IF ((A[I,3]=7)AND(SAYI<>0))OR((A[I,3]<7)AND(SAYI=4)) THEN
        BEGIN YARDIM:=19;HELP(YARDIM,HELPX);GOTO 3 END;
       END;
  8..23:IF A[I,2+2*GIRIS]<>0 THEN
        BEGIN YARDIM:=20;HELP(YARDIM,HELPX);GOTO 3 END;
 END;
 IF ((A[PW,3]<8)OR((A[PW,3]>11)AND(A[PW,3]<22)))AND
    ((A[I,3]<8)OR((A[I,3]>11)AND(A[I,3]<22))) THEN
 BEGIN
  IF PW=I THEN BEGIN YARDIM:=6;HELP(YARDIM,HELPX);GOTO 3 END;
  FOR I1:=1 TO 180 DO FOR I2:=1 TO 2 DO C[I1,I2]:=0;
  I1:=0;
  FOR I2:=1 TO SON1 DO
  BEGIN
   CASE A[I2,3] OF
      1..7:BEGIN
            IF A[I2,5]<>0 THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,5];C[I1,2]:=I2 END;
            IF A[I2,7]<>0 THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,7];C[I1,2]:=I2 END;
            IF A[I2,9]<>0 THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,9];C[I1,2]:=I2 END;
```

```
            IF A[I2,11]<>0 THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,11];C[I1,2]:=I2 END;
           END;
   12..21:BEGIN
            IF (A[I2,4]>0)AND(A[I2,4]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,4];C[I1,2]:=I2 END;
            IF (A[I2,6]>0)AND(A[I2,6]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,6];C[I1,2]:=I2 END;
            IF (A[I2,8]>0)AND(A[I2,8]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,8];C[I1,2]:=I2 END;
            IF (A[I2,10]>0)AND(A[I2,10]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,10];C[I1,2]:=I2 END;
            IF (A[I2,12]>0)AND(A[I2,12]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,12];C[I1,2]:=I2 END;
            IF (A[I2,14]>0)AND(A[I2,14]<31) THEN
            BEGIN I1:=I1+1;C[I1,1]:=A[I2,14];C[I1,2]:=I2 END;
           END;
  END;
 END;
 1:I2:=PW;I3:=0;I4:=0;
 2:I1:=0;I3:=I3+1;
 REPEAT I1:=I1+1 UNTIL (C[I1,2]=I2)OR(I1=180);
 IF I1=180 THEN
 IF I3=1 THEN GOTO 4 ELSE BEGIN C[I4,1]:=0;C[I4,2]:=0;GOTO 1 END;
 I4:=I1;
 IF C[I1,1]=I THEN BEGIN YARDIM:=6;HELP(YARDIM,HELPX);GOTO 3 END
 ELSE BEGIN I2:=C[I1,1];GOTO 2 END;
END;
4:CASE A[I,3] OF
  1..7:BEGIN
        SAYI:=A[I,4]+A[I,13];
        IF ((A[I,3]=7)AND(SAYI=0))OR((A[I,3]<7)AND(SAYI<4)) THEN
        BEGIN
         A[I,4]:=A[I,4]+1;A[I,3+2*A[I,4]]:=PW;A[I,4+2*A[I,4]]:=CIKIS
        END;
       END;
 8..23:IF A[I,2+2*GIRIS]=0 THEN
```

```
        BEGIN A[I,2+2*GIRIS]:=PW;A[I,3+2*GIRIS]:=CIKIS END;
 END;
 3:IF (YARDIM=19)OR(YARDIM=20)OR(YARDIM=6) THEN
 BEGIN
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  YARDIM:=5;HELP(YARDIM,HELPX);PUT;
 END ELSE BEGIN PUT;REPEAT UNTIL MOUSEBUTTONS=EVNOTHING END;
END;
PROCEDURE TAKE_WIRE;
VAR TW:INTEGER;
BEGIN
 MARK_OUTPUT;IF PT=1 THEN EXIT;ISARET:=1;TW:=I;
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 REPEAT MOUSE;MARK_INPUT UNTIL PT=0;
 ISARET:=0;
 CASE A[I,3] OF
  1..7:BEGIN
     IF A[I,4]=0 THEN BEGIN YARDIM:=8;HELP(YARDIM,HELPX) END
     ELSE IF A[I,11]=TW THEN
     BEGIN A[I,4]:=A[I,4]-1;A[I,11]:=0;A[I,12]:=0 END
     ELSE IF A[I,9]=TW THEN
     BEGIN
      A[I,4]:=A[I,4]-1;A[I,9]:=A[I,11];A[I,10]:=A[I,12];
      A[I,11]:=0;A[I,12]:=0
     END
     ELSE IF A[I,7]=TW THEN
     BEGIN
      A[I,4]:=A[I,4]-1;A[I,7]:=A[I,9];A[I,8]:=A[I,10];
      A[I,9]:=A[I,11];A[I,10]:=A[I,12];A[I,11]:=0;A[I,12]:=0
     END
     ELSE IF A[I,5]=TW THEN
     BEGIN
      A[I,4]:=A[I,4]-1;A[I,5]:=A[I,7];A[I,6]:=A[I,8];
      A[I,7]:=A[I,9];A[I,8]:=A[I,10];A[I,9]:=A[I,11];
      A[I,10]:=A[I,12];A[I,11]:=0;A[I,12]:=0
```

```
     END ELSE BEGIN YARDIM:=8;HELP(YARDIM,HELPX) END;
    END;
  8..23:IF (A[I,2+2*GIRIS]=0)OR(A[I,2+2*GIRIS]>30) THEN
        BEGIN YARDIM:=8;HELP(YARDIM,HELPX) END ELSE
        IF A[I,2+2*GIRIS]=TW THEN
        BEGIN A[I,2+2*GIRIS]:=0;A[I,3+2*GIRIS]:=0 END
        ELSE BEGIN YARDIM:=8;HELP(YARDIM,HELPX) END;
 END;
 IF YARDIM=8 THEN
 BEGIN
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  YARDIM:=7;HELP(YARDIM,HELPX);PUT;
 END ELSE BEGIN PUT;REPEAT UNTIL MOUSEBUTTONS=EVNOTHING END;
END;
PROCEDURE PUT_INPUT;
BEGIN
 MARK_INPUT;IF PT=1 THEN EXIT;YARDIM:=10;HELP(YARDIM,HELPX);
 HIDEMOUSE;WHILE KEYPRESSED DO CH:=UPCASE(READKEY);CH:=' ';
 REPEAT
  IF KEYPRESSED THEN CH:=UPCASE(READKEY)
 UNTIL (CH='X')OR(CH='Y')OR(CH='Z')OR(CH='W')OR(CH='0')OR(CH='1');
 SHOWMOUSE;YARDIM:=9;HELP(YARDIM,HELPX);
 CASE CH OF
  'X':H:=31;'Y':H:=32;'Z':H:=33;'W':H:=34;'0':H:=35;'1':H:=36
 END;
 CASE GIRIS OF
  0:BEGIN
     SAYI:=A[I,4]+A[I,13];
     IF ((A[I,3]=7)AND(SAYI=0))OR((A[I,3]<7)AND(SAYI<4)) THEN
     BEGIN A[I,13]:=A[I,13]+1;A[I,13+A[I,13]]:=H END
     ELSE BEGIN YARDIM:=19;HELP(YARDIM,HELPX) END;
    END;
  1..6:IF A[I,2+2*GIRIS]=0 THEN
    BEGIN A[I,2+2*GIRIS]:=H;A[I,3+2*GIRIS]:=0 END
    ELSE BEGIN YARDIM:=20;HELP(YARDIM,HELPX) END;
```

```
 END;
 IF (YARDIM=19)OR(YARDIM=20) THEN
 BEGIN
  REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  YARDIM:=9;HELP(YARDIM,HELPX);
 END;
 PUT;
END;
PROCEDURE TAKE_INPUT;
BEGIN
 MARK_INPUT;IF PT=1 THEN EXIT;YARDIM:=13;HELP(YARDIM,HELPX);
 HIDEMOUSE;WHILE KEYPRESSED DO CH:=UPCASE(READKEY);CH:=' ';
 REPEAT
  IF KEYPRESSED THEN CH:=UPCASE(READKEY)
 UNTIL (CH='X')OR(CH='Y')OR(CH='Z')OR(CH='W')OR(CH='0')OR(CH='1');
 SHOWMOUSE;YARDIM:=11;HELP(YARDIM,HELPX);
 CASE CH OF
  'X':H:=31;'Y':H:=32;'Z':H:=33;'W':H:=34;'0':H:=35;'1':H:=36
 END;
 CASE GIRIS OF
  0:BEGIN
     IF A[I,13]=0 THEN BEGIN YARDIM:=12;HELP(YARDIM,HELPX) END
     ELSE IF A[I,17]=H THEN BEGIN A[I,13]:=A[I,13]-1;A[I,17]:=0 END
     ELSE IF A[I,16]=H THEN
     BEGIN A[I,13]:=A[I,13]-1;A[I,16]:=A[I,17];A[I,17]:=0 END
     ELSE IF A[I,15]=H THEN
     BEGIN
      A[I,13]:=A[I,13]-1;A[I,15]:=A[I,16];A[I,16]:=A[I,17];A[I,17]:=0
     END
     ELSE IF A[I,14]=H THEN
     BEGIN
      A[I,13]:=A[I,13]-1;A[I,14]:=A[I,15];A[I,15]:=A[I,16];
      A[I,16]:=A[I,17];A[I,17]:=0
     END
     ELSE BEGIN YARDIM:=14;HELP(YARDIM,HELPX) END;
    END;
```

```
 1..6:IF A[I,2+2*GIRIS]<31 THEN
      BEGIN YARDIM:=21;HELP(YARDIM,HELPX) END ELSE
      IF A[I,2+2*GIRIS]=H THEN
      BEGIN A[I,2+2*GIRIS]:=0;A[I,3+2*GIRIS]:=0 END
      ELSE BEGIN YARDIM:=22;HELP(YARDIM,HELPX) END;
 END;
 IF (YARDIM=12)OR(YARDIM=21)OR(YARDIM=14)OR(YARDIM=22) THEN
 BEGIN
  REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  YARDIM:=11;HELP(YARDIM,HELPX);
 END;
 PUT;
END;
PROCEDURE FOPART1_7;
BEGIN
 CASE O[HASANT,4]+O[HASANT,13] OF
  0:B[HASANT,1]:=60;
  1:CASE O[HASANT,3] OF
     1..6:B[HASANT,1]:=60;
     7:CASE O[HASANT,4] OF
        0:BEGIN B[HASANT,1]:=O[HASANT,14];B[HASANT,2]:=47 END;
        1:CASE O[O[HASANT,5],3] OF
           1..7,12..21:BEGIN
            B[HASANT,1]:=O[HASANT,5];
            B[HASANT,2]:=O[HASANT,6];
            B[HASANT,3]:=47
           END;
           8..11:BEGIN
            B[HASANT,1]:=O[O[HASANT,5],20]+100;
            IF O[HASANT,6]=-1 THEN B[HASANT,2]:=47
           END;
           22,23:BEGIN
            B[HASANT,1]:=O[O[HASANT,5],20]+99-O[HASANT,6];
            B[HASANT,2]:=47
           END;
        END;
```

```
   END;
END;
2,3,4:BEGIN
       B[HASANT,1]:=51;G2:=2;G3:=3;
       WHILE G3<3+2*O[HASANT,4] DO
       BEGIN
        G3:=G3+2;
        CASE O[O[HASANT,G3],3] OF
         1..7,12..21:BEGIN
           B[HASANT,G2]:=O[HASANT,G3];
           B[HASANT,G2+1]:=O[HASANT,G3+1];
           B[HASANT,G2+2]:=O[HASANT,3]+40;
           G2:=G2+3;
         END;
         8..11:BEGIN
          B[HASANT,G2]:=O[O[HASANT,G3],20]+100;
          IF O[HASANT,G3+1]=-1 THEN
          BEGIN
           B[HASANT,G2+1]:=O[HASANT,3]+40;G2:=G2+2;
          END ELSE
          BEGIN
           B[HASANT,G2+1]:=47;
           B[HASANT,G2+2]:=O[HASANT,3]+40;G2:=G2+3;
          END;
         END;
         22,23:BEGIN
          B[HASANT,G2]:=O[O[HASANT,G3],20]+99-O[HASANT,G3+1];
          B[HASANT,G2+1]:=O[HASANT,3]+40;G2:=G2+2;
       END END END;
       FOR G3:=14 TO 17 DO IF O[HASANT,G3]>30 THEN
       BEGIN
        B[HASANT,G2]:=O[HASANT,G3];
        B[HASANT,G2+1]:=O[HASANT,3]+40;G2:=G2+2
       END;
       B[HASANT,G2-1]:=52;
       IF (O[HASANT,3]=2)OR(O[HASANT,3]=4) THEN B[HASANT,G2]:=47;
      END;
```

```
  END
END;
PROCEDURE TEMURTAS1;
BEGIN
 IF O[HASANT,4]>30 THEN BEGIN HT11:=O[HASANT,4];HT12:=0 END ELSE
  CASE O[O[HASANT,4],3] OF
   1..7,12..21:BEGIN HT11:=O[HASANT,4];HT12:=O[HASANT,5] END;
   8..11:BEGIN HT11:=O[O[HASANT,4],20]+100;
    IF O[HASANT,5]=-1 THEN HT12:=0 ELSE HT12:=47;
   END;
   22,23:BEGIN HT11:=O[O[HASANT,4],20]+99-O[HASANT,5];HT12:=0 END;
  END;
END;
PROCEDURE TEMURTAS2;
BEGIN
 IF O[HASANT,6]>30 THEN BEGIN HT21:=O[HASANT,6];HT22:=0 END ELSE
  CASE O[O[HASANT,6],3] OF
   1..7,12..21:BEGIN HT21:=O[HASANT,6];HT22:=O[HASANT,7] END;
   8..11:BEGIN HT21:=O[O[HASANT,6],20]+100;
    IF O[HASANT,7]=-1 THEN HT22:=0 ELSE HT22:=47;
   END;
   22,23:BEGIN HT21:=O[O[HASANT,6],20]+99-O[HASANT,7];HT22:=0 END;
  END;
END;
PROCEDURE TEMURTAS3;
BEGIN
 IF O[HASANT,8]>30 THEN BEGIN HT31:=O[HASANT,8];HT32:=0 END ELSE
  CASE O[O[HASANT,8],3] OF
   1..7,12..21:BEGIN HT31:=O[HASANT,8];HT32:=O[HASANT,9] END;
   8..11:BEGIN HT31:=O[O[HASANT,8],20]+100;
    IF O[HASANT,9]=-1 THEN HT32:=0 ELSE HT32:=47;
   END;
   22,23:BEGIN HT31:=O[O[HASANT,8],20]+99-O[HASANT,9];HT32:=0 END;
  END;
END;
```

```
PROCEDURE TEMURTAS4;
BEGIN
 IF Q[HASANT,10]>30 THEN BEGIN HT41:=O[HASANT,10];HT42:=0 END ELSE
  CASE O[O[HASANT,10],3] OF
   1..7,12..21:BEGIN HT41:=O[HASANT,10];HT42:=O[HASANT,11] END;
   8..11:BEGIN HT41:=O[O[HASANT,10],20]+100;
    IF O[HASANT,11]=-1 THEN HT42:=0 ELSE HT42:=47;
   END;
   22,23:BEGIN HT41:=O[O[HASANT,10],20]+99-O[HASANT,11];HT42:=0 END;
  END;
END;
PROCEDURE TEMURTAS5;
BEGIN
 IF O[HASANT,12]>30 THEN BEGIN HT51:=O[HASANT,12];HT52:=0 END ELSE
  CASE O[O[HASANT,12],3] OF
   1..7,12..21:BEGIN HT51:=O[HASANT,12];HT52:=O[HASANT,13] END;
   8..11:BEGIN HT51:=O[O[HASANT,12],20]+100;
    IF O[HASANT,13]=-1 THEN HT52:=0 ELSE HT52:=47;
   END;
   22,23:BEGIN HT51:=O[O[HASANT,12],20]+99-O[HASANT,13];HT52:=0 END;
  END;
END;
PROCEDURE TEMURTAS6;
BEGIN
 IF O[HASANT,14]>30 THEN BEGIN HT61:=O[HASANT,14];HT62:=0 END ELSE
  CASE O[O[HASANT,14],3] OF
   1..7,12..21:BEGIN HT61:=O[HASANT,14];HT62:=O[HASANT,15] END;
   8..11:BEGIN HT61:=O[O[HASANT,14],20]+100;
    IF O[HASANT,15]=-1 THEN HT62:=0 ELSE HT62:=47;
   END;
   22,23:BEGIN HT61:=O[O[HASANT,14],20]+99-O[HASANT,15];HT62:=0 END;
  END;
END;
PROCEDURE FOPART8;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
```

```
  TEMURTAS1;TEMURTAS2;
  FOR HTEMUR:=1 TO 2 DO
  BEGIN
   IF HTEMUR=1 THEN TEMUR:=1 ELSE TEMUR:=51;
   B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=51;B[HASANT,TEMUR+2]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+3]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=O[HASANT,20]+100;
   B[HASANT,TEMUR+5]:=47;B[HASANT,TEMUR+6]:=52;B[HASANT,TEMUR+7]:=43;
   B[HASANT,TEMUR+8]:=51;B[HASANT,TEMUR+9]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+10]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+10]:=47;
   B[HASANT,TEMUR+11]:=41;B[HASANT,TEMUR+12]:=O[HASANT,20]+100;
   B[HASANT,TEMUR+13]:=52;B[HASANT,TEMUR+14]:=52;
   IF HTEMUR=2 THEN B[HASANT,TEMUR+15]:=47;
  END;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART9;
BEGIN
 IF O[HASANT,4]<>0 THEN
 BEGIN
  TEMURTAS1;
  FOR HTEMUR:=0 TO 1 DO
  BEGIN
   B[HASANT,50*HTEMUR+1]:=HT11;
   IF HT12<>0 THEN B[HASANT,50*HTEMUR+2]:=HT12;
   IF HTEMUR=1 THEN
   IF HT12<>0 THEN B[HASANT,50*HTEMUR+3]:=47
   ELSE B[HASANT,50*HTEMUR+2]:=47;
  END;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART10;
BEGIN
 IF O[HASANT,4]<>0 THEN
 BEGIN
  TEMURTAS1;
```

```
   FOR HTEMUR:=1 TO 2 DO
   BEGIN
    IF HTEMUR=1 THEN TEMUR:=1 ELSE TEMUR:=51;
    B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=51;B[HASANT,TEMUR+2]:=HT11;
    IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+3]:=HT12;TEMUR:=TEMUR+1 END;
    B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=O[HASANT,20]+100;
    B[HASANT,TEMUR+5]:=47;B[HASANT,TEMUR+6]:=52;
    B[HASANT,TEMUR+7]:=43;B[HASANT,TEMUR+8]:=51;
    B[HASANT,TEMUR+9]:=HT11;
    IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+10]:=HT12;TEMUR:=TEMUR+1 END;
    B[HASANT,TEMUR+10]:=47;
    B[HASANT,TEMUR+11]:=41;B[HASANT,TEMUR+12]:=O[HASANT,20]+100;
    B[HASANT,TEMUR+13]:=52;B[HASANT,TEMUR+14]:=52;
    IF HTEMUR=2 THEN B[HASANT,TEMUR+15]:=47;
   END;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART11;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;
  FOR HTEMUR:=1 TO 2 DO
  BEGIN
   IF HTEMUR=1 THEN TEMUR:=1 ELSE TEMUR:=51;
   B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+2]:=43;B[HASANT,TEMUR+3]:=51;
   B[HASANT,TEMUR+4]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+5]:=47;
   B[HASANT,TEMUR+6]:=41;B[HASANT,TEMUR+7]:=O[HASANT,20]+100;
   B[HASANT,TEMUR+8]:=52;B[HASANT,TEMUR+9]:=52;
   IF HTEMUR=2 THEN B[HASANT,TEMUR+10]:=47;
  END;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
```

```
PROCEDURE FOPART12;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;
  TEMUR:=1;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=45;B[HASANT,TEMUR+3]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=52;
  TEMUR:=51;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=52;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART13;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0)AND(O[HASANT,8]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;TEMURTAS3;
  TEMUR:=1;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=45;B[HASANT,TEMUR+3]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=45;B[HASANT,TEMUR+5]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+6]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+6]:=52;
  TEMUR:=51;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=51;
  B[HASANT,TEMUR+2]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+3]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT22;TEMUR:=TEMUR+1 END;
```

```
   B[HASANT,TEMUR+5]:=52;B[HASANT,TEMUR+6]:=43;
   B[HASANT,TEMUR+7]:=51;
   B[HASANT,TEMUR+8]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+9]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+9]:=41;B[HASANT,TEMUR+10]:=HT31;
   IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+11]:=HT32;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+11]:=52;B[HASANT,TEMUR+12]:=43;
   B[HASANT,TEMUR+13]:=51;
   B[HASANT,TEMUR+14]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+15]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+15]:=41;B[HASANT,TEMUR+16]:=HT31;
   IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+17]:=HT32;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+17]:=52;B[HASANT,TEMUR+18]:=52;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART14;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;
  B[HASANT,1]:=HT21;
  IF HT22<>0 THEN B[HASANT,2]:=HT22;
 END ELSE B[HASANT,1]:=60;
END;
PROCEDURE FOPART15;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0)AND(O[HASANT,8]<>0)AND
    (O[HASANT,10]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;TEMURTAS3;TEMURTAS4;
  TEMUR:=1;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=43;B[HASANT,TEMUR+3]:=HT41;
  IF HT42<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT42;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=52;
  TEMUR:=51;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
```

```
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=43;B[HASANT,TEMUR+3]:=HT41;
  IF HT42<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT42;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=52;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART16;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0)AND(O[HASANT,8]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;TEMURTAS3;
  TEMUR:=1;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=51;
  B[HASANT,TEMUR+2]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+3]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+3]:=47;
  B[HASANT,TEMUR+4]:=41;B[HASANT,TEMUR+5]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+6]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+6]:=52;B[HASANT,TEMUR+7]:=43;
  B[HASANT,TEMUR+8]:=51;B[HASANT,TEMUR+9]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+10]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+10]:=41;B[HASANT,TEMUR+11]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+12]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+12]:=52;B[HASANT,TEMUR+13]:=52;
 END ELSE B[HASANT,1]:=60;
END;
PROCEDURE FOPART17;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0)AND(O[HASANT,8]<>0)AND
    (O[HASANT,10]<>0)AND(O[HASANT,12]<>0)AND(O[HASANT,14]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;TEMURTAS3;TEMURTAS4;TEMURTAS5;TEMURTAS6;
  TEMUR:=1;B[HASANT,TEMUR]:=51;
  B[HASANT,TEMUR+1]:=51;B[HASANT,TEMUR+2]:=HT51;
  IF HT52<>0 THEN BEGIN B[HASANT,TEMUR+3]:=HT52;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+3]:=47;B[HASANT,TEMUR+4]:=41;B[HASANT,TEMUR+5]:=HT61;
  IF HT62<>0 THEN BEGIN B[HASANT,TEMUR+6]:=HT62;TEMUR:=TEMUR+1 END;
```

```
   B[HASANT,TEMUR+6]:=47;B[HASANT,TEMUR+7]:=41;B[HASANT,TEMUR+8]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+9]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+9]:=52;B[HASANT,TEMUR+10]:=43;
   B[HASANT,TEMUR+11]:=51;B[HASANT,TEMUR+12]:=HT51;
   IF HT52<>0 THEN BEGIN B[HASANT,TEMUR+13]:=HT52;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+13]:=47;B[HASANT,TEMUR+14]:=41;
   B[HASANT,TEMUR+15]:=HT61;
   IF HT62<>0 THEN BEGIN B[HASANT,TEMUR+16]:=HT62;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+16]:=41;B[HASANT,TEMUR+17]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+18]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+18]:=52;B[HASANT,TEMUR+19]:=43;
   B[HASANT,TEMUR+20]:=51;B[HASANT,TEMUR+21]:=HT51;
   IF HT52<>0 THEN BEGIN B[HASANT,TEMUR+22]:=HT52;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+22]:=41;B[HASANT,TEMUR+23]:=HT61;
   IF HT62<>0 THEN BEGIN B[HASANT,TEMUR+24]:=HT62;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+24]:=47;B[HASANT,TEMUR+25]:=41;
   B[HASANT,TEMUR+26]:=HT31;
   IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+27]:=HT32;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+27]:=52;B[HASANT,TEMUR+28]:=43;
   B[HASANT,TEMUR+29]:=51;B[HASANT,TEMUR+30]:=HT51;
   IF HT52<>0 THEN BEGIN B[HASANT,TEMUR+31]:=HT52;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+31]:=41;B[HASANT,TEMUR+32]:=HT61;
   IF HT62<>0 THEN BEGIN B[HASANT,TEMUR+33]:=HT62;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+33]:=41;B[HASANT,TEMUR+34]:=HT41;
   IF HT42<>0 THEN BEGIN B[HASANT,TEMUR+35]:=HT42;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+35]:=52;B[HASANT,TEMUR+36]:=52;
 END ELSE B[HASANT,1]:=60;
END;
PROCEDURE FOPART18;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;
  TEMUR:=1;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=47;B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT11;
```

```
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+5]:=52;
  TEMUR:=51;
  B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=52;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART19;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0)AND(O[HASANT,8]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;TEMURTAS3;
  TEMUR:=1;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=47;B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+5]:=47;B[HASANT,TEMUR+6]:=41;B[HASANT,TEMUR+7]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+8]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+8]:=52;
  TEMUR:=51;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=47;B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+5]:=41;B[HASANT,TEMUR+6]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+7]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+7]:=52;
  TEMUR:=101;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=47;B[HASANT,TEMUR+5]:=41;
  B[HASANT,TEMUR+6]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+7]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+7]:=52;
```

```
  TEMUR:=151;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT31;
  IF HT32<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT32;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+4]:=41;
  B[HASANT,TEMUR+5]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+6]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+6]:=52;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60;
 B[HASANT,101]:=60;B[HASANT,151]:=60 END;
END;
PROCEDURE FOPART20;
BEGIN
 IF O[HASANT,4]<>0 THEN
 BEGIN
  TEMURTAS1;
  TEMUR:=1;B[HASANT,TEMUR]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+1]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+1]:=47;
  TEMUR:=51;B[HASANT,TEMUR]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+1]:=HT12;TEMUR:=TEMUR+1 END;
 END ELSE BEGIN B[HASANT,1]:=60;B[HASANT,51]:=60 END;
END;
PROCEDURE FOPART21;
BEGIN
 IF (O[HASANT,4]<>0)AND(O[HASANT,6]<>0) THEN
 BEGIN
  TEMURTAS1;TEMURTAS2;
  TEMUR:=1;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=47;B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT22;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+5]:=47;B[HASANT,TEMUR+6]:=52;
  TEMUR:=51;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
  IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
  B[HASANT,TEMUR+2]:=47;B[HASANT,TEMUR+3]:=41;B[HASANT,TEMUR+4]:=HT21;
  IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+5]:=HT22;TEMUR:=TEMUR+1 END;
```

```
   B[HASANT,TEMUR+5]:=52;
   TEMUR:=101;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+4]:=47;B[HASANT,TEMUR+5]:=52;
   TEMUR:=151;B[HASANT,TEMUR]:=51;B[HASANT,TEMUR+1]:=HT11;
   IF HT12<>0 THEN BEGIN B[HASANT,TEMUR+2]:=HT12;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+2]:=41;B[HASANT,TEMUR+3]:=HT21;
   IF HT22<>0 THEN BEGIN B[HASANT,TEMUR+4]:=HT22;TEMUR:=TEMUR+1 END;
   B[HASANT,TEMUR+4]:=52;
 END ELSE
 BEGIN
   B[HASANT,1]:=60;B[HASANT,51]:=60;B[HASANT,101]:=60;
   B[HASANT,151]:=60
 END;
END;
PROCEDURE FOPART22;
BEGIN
 IF O[HASANT,4]<>0 THEN
 BEGIN
  TEMURTAS1;B[HASANT,1]:=HT11;IF HT12<>0 THEN B[HASANT,2]:=HT12;
 END ELSE B[HASANT,1]:=60;
 IF O[HASANT,6]<>0 THEN
 BEGIN
  TEMURTAS2;B[HASANT,51]:=HT21;IF HT22<>0 THEN B[HASANT,52]:=HT22;
 END ELSE B[HASANT,51]:=60;
END;
PROCEDURE FOPART23;
BEGIN
 IF O[HASANT,4]<>0 THEN
 BEGIN
  TEMURTAS1;B[HASANT,1]:=HT11;IF HT12<>0 THEN B[HASANT,2]:=HT12;
 END ELSE B[HASANT,1]:=60;
 IF O[HASANT,6]<>0 THEN
 BEGIN
  TEMURTAS2;B[HASANT,51]:=HT21;IF HT22<>0 THEN B[HASANT,52]:=HT22;
```

```
 END ELSE B[HASANT,51]:=60;
 IF O[HASANT,8]<>0 THEN
 BEGIN
  TEMURTAS3;B[HASANT,101]:=HT31;IF HT32<>0 THEN B[HASANT,102]:=HT32;
 END ELSE B[HASANT,101]:=60;
 IF O[HASANT,10]<>0 THEN
 BEGIN
  TEMURTAS4;B[HASANT,151]:=HT41;IF HT42<>0 THEN B[HASANT,152]:=HT42;
 END ELSE B[HASANT,151]:=60;
END;
PROCEDURE REDUCTION_ZEROS_AND_ONES;
VAR AC,KAPA:INTEGER;
BEGIN
 REPEAT
  G:=0;REPEAT G:=G+1 UNTIL (Q1[G]=0)OR(Q1[G]=35)OR(Q1[G]=36);
  IF Q1[G]=0 THEN G1:=0 ELSE
  BEGIN
   IF Q1[G]=35 THEN
   BEGIN
    IF Q1[G+1]=0 THEN G1:=0;
    IF (Q1[G+1]=41)OR(Q1[G-1]=41) THEN G1:=1;
    IF (Q1[G+1]=43)OR(Q1[G-1]=43) THEN G1:=2;
    IF (Q1[G+1]=45)OR(Q1[G-1]=45) THEN G1:=3;
    IF (Q1[G+1]=46)OR(Q1[G-1]=46) THEN G1:=4;
    IF Q1[G+1]=47 THEN G1:=9;
    IF (Q1[G+1]=52)AND(Q1[G-1]=51) THEN G1:=10;
   END;
   IF Q1[G]=36 THEN
   BEGIN
    IF Q1[G+1]=0 THEN G1:=0;
    IF (Q1[G+1]=41)OR(Q1[G-1]=41) THEN G1:=5;
    IF (Q1[G+1]=43)OR(Q1[G-1]=43) THEN G1:=6;
    IF (Q1[G+1]=45)OR(Q1[G-1]=45) THEN G1:=7;
    IF (Q1[G+1]=46)OR(Q1[G-1]=46) THEN G1:=8;
    IF Q1[G+1]=47 THEN G1:=9;
    IF (Q1[G+1]=52)AND(Q1[G-1]=51) THEN G1:=10;
   END;
```

```
END;
IF G1=1 THEN
BEGIN
 IF Q1[G+1]=41 THEN
 BEGIN
  IF Q1[G+2]<>51 THEN
  BEGIN
   IF Q1[G+3]<>47 THEN
   BEGIN
    G:=G+2;REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0;
   END ELSE
   BEGIN G:=G+3;REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0 END;
  END ELSE
  BEGIN
   AC:=1;KAPA:=0;G2:=G+2;
   REPEAT
    G2:=G2+1;IF Q1[G2]=51 THEN AC:=AC+1;
    IF Q1[G2]=52 THEN KAPA:=KAPA+1;
   UNTIL AC=KAPA;
   IF Q1[G2+1]=47 THEN G2:=G2+1;
   REPEAT G2:=G2+1;G:=G+1;Q1[G]:=Q1[G2] UNTIL Q1[G2]=0;
 END;
END ELSE IF Q1[G-1]=41 THEN
BEGIN
 IF Q1[G-2]<>47 THEN
 BEGIN
  IF Q1[G-2]<>52 THEN
  REPEAT G:=G+1;Q1[G-3]:=Q1[G-1] UNTIL Q1[G-1]=0
  ELSE
  BEGIN
   AC:=0;KAPA:=1;G2:=G-2;
   REPEAT
    G2:=G2-1;IF Q1[G2]=51 THEN AC:=AC+1;
    IF Q1[G2]=52 THEN KAPA:=KAPA+1;
   UNTIL AC=KAPA;
   REPEAT G2:=G2+1;G:=G+1;Q1[G2-1]:=Q1[G-1] UNTIL Q1[G-1]=0;
  END;
```

```
  END ELSE
  BEGIN
   IF Q1[G-3]<>52 THEN
   REPEAT G:=G+1;Q1[G-4]:=Q1[G-1] UNTIL Q1[G-1]=0
   ELSE
   BEGIN
    AC:=0;KAPA:=1;G2:=G-3;
    REPEAT
     G2:=G2-1;IF Q1[G2]=51 THEN AC:=AC+1;
     IF Q1[G2]=52 THEN KAPA:=KAPA+1;
    UNTIL AC=KAPA;
    REPEAT G2:=G2+1;G:=G+1;Q1[G2-1]:=Q1[G-1] UNTIL Q1[G-1]=0;
END END END END;
IF G1=2 THEN
BEGIN
 IF Q1[G+1]=43 THEN REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL Q1[G+1]=0
 ELSE IF Q1[G-1]=43 THEN REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
END;
IF G1=3 THEN
BEGIN
 IF Q1[G+1]=45 THEN REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL Q1[G+1]=0
 ELSE IF Q1[G-1]=45 THEN REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
END;
IF G1=4 THEN
BEGIN
 IF Q1[G+1]=46 THEN
 BEGIN
  IF Q1[G+2]<>51 THEN
  BEGIN
   IF Q1[G+3]<>47 THEN
   BEGIN
    Q1[G]:=Q1[G+2];Q1[G+1]:=47;G:=G+2;
    REPEAT G:=G+1;Q1[G-1]:=Q1[G] UNTIL Q1[G]=0;
   END ELSE
   BEGIN
    Q1[G]:=Q1[G+2];G:=G+3;
    REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0;
```

```
   END;
  END ELSE
  BEGIN
   AC:=1;KAPA:=0;G2:=G+2;
   REPEAT
    G2:=G2+1;IF Q1[G2]=51 THEN AC:=AC+1;
    IF Q1[G2]=52 THEN KAPA:=KAPA+1;
   UNTIL AC=KAPA;
   IF Q1[G2+1]<>47 THEN
   BEGIN
    REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL G=G2-1;
    Q1[G]:=47;
    REPEAT G:=G+1;Q1[G]:=Q1[G+1] UNTIL Q1[G+1]=0;
   END ELSE
   BEGIN
    REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL G=G2-1;
    REPEAT G:=G+1;Q1[G-1]:=Q1[G+2] UNTIL Q1[G+2]=0;
   END;
  END;
 END ELSE IF Q1[G-1]=46 THEN
 BEGIN
  IF Q1[G-2]<>47 THEN
  BEGIN
   Q1[G-1]:=47;
   REPEAT G:=G+1;Q1[G-1]:=Q1[G] UNTIL Q1[G]=0;
  END ELSE REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0;
 END;
END;
IF G1=5 THEN
BEGIN
 IF Q1[G+1]=41 THEN REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL Q1[G+1]=0
 ELSE IF Q1[G-1]=41 THEN REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
END;
IF G1=6 THEN
BEGIN
 IF Q1[G+1]=43 THEN
 BEGIN
```

```
 IF Q1[G+2]<>51 THEN
 BEGIN
  IF Q1[G+3]<>47 THEN
  BEGIN
   G:=G+2;REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0;
  END ELSE
  BEGIN
   G:=G+3;REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0;
  END;
 END ELSE
 BEGIN
  AC:=1;KAPA:=0;G2:=G+2;
  REPEAT
   G2:=G2+1;IF Q1[G2]=51 THEN AC:=AC+1;
   IF Q1[G2]=52 THEN KAPA:=KAPA+1;
  UNTIL AC=KAPA;
  IF Q1[G2+1]=47 THEN G2:=G2+1;
  REPEAT G2:=G2+1;G:=G+1;Q1[G]:=Q1[G2] UNTIL Q1[G2]=0;
 END;
END ELSE IF Q1[G-1]=43 THEN
BEGIN
 IF Q1[G-2]<>47 THEN
 BEGIN
  IF Q1[G-2]<>52 THEN
  REPEAT G:=G+1;Q1[G-3]:=Q1[G-1] UNTIL Q1[G-1]=0
  ELSE
  BEGIN
   AC:=0;KAPA:=1;G2:=G-2;
   REPEAT
    G2:=G2-1;IF Q1[G2]=51 THEN AC:=AC+1;
    IF Q1[G2]=52 THEN KAPA:=KAPA+1;
   UNTIL AC=KAPA;
   REPEAT G2:=G2+1;G:=G+1;Q1[G2-1]:=Q1[G-1] UNTIL Q1[G-1]=0;
  END;
 END ELSE
 BEGIN
  IF Q1[G-3]<>52 THEN
```

```
    REPEAT G:=G+1;Q1[G-4]:=Q1[G-1] UNTIL Q1[G-1]=0
    ELSE
    BEGIN
     AC:=0;KAPA:=1;G2:=G-3;
     REPEAT
      G2:=G2-1;IF Q1[G2]=51 THEN AC:=AC+1;
      IF Q1[G2]=52 THEN KAPA:=KAPA+1;
     UNTIL AC=KAPA;
     REPEAT G2:=G2+1;G:=G+1;Q1[G2-1]:=Q1[G-1] UNTIL Q1[G-1]=0;
END END END END;
IF G1=7 THEN
BEGIN
 IF Q1[G+1]=45 THEN
 BEGIN
  IF Q1[G+2]<>51 THEN
  BEGIN
   IF Q1[G+3]<>47 THEN
   BEGIN
    Q1[G]:=Q1[G+2];Q1[G+1]:=47;G:=G+2;
    REPEAT G:=G+1;Q1[G-1]:=Q1[G] UNTIL Q1[G]=0;
   END ELSE
   BEGIN
    Q1[G]:=Q1[G+2];G:=G+3;
    REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0;
   END;
  END ELSE
  BEGIN
   AC:=1;KAPA:=0;G2:=G+2;
   REPEAT
    G2:=G2+1;IF Q1[G2]=51 THEN AC:=AC+1;
    IF Q1[G2]=52 THEN KAPA:=KAPA+1;
   UNTIL AC=KAPA;
   IF Q1[G2+1]<>47 THEN
   BEGIN
    REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL (G+1)=G2;
    Q1[G]:=47;
    REPEAT G:=G+1;Q1[G]:=Q1[G+1] UNTIL Q1[G+1]=0;
```

```
      END ELSE
      BEGIN
       REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL (G+1)=G2;
       REPEAT G:=G+1;Q1[G-1]:=Q1[G+2] UNTIL Q1[G+2]=0;
      END;
     END;
    END ELSE IF Q1[G-1]=45 THEN
    BEGIN
     IF Q1[G-2]<>47 THEN
     BEGIN
      Q1[G-1]:=47;
      REPEAT G:=G+1;Q1[G-1]:=Q1[G] UNTIL Q1[G]=0;
     END ELSE REPEAT G:=G+1;Q1[G-3]:=Q1[G] UNTIL Q1[G]=0;
    END;
   END;
   IF G1=8 THEN
   BEGIN
    IF Q1[G+1]=46 THEN REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL Q1[G+1]=0
    ELSE IF Q1[G-1]=46 THEN REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
   END;
   IF G1=9 THEN
   BEGIN
    IF Q1[G]=35 THEN Q1[G]:=36 ELSE Q1[G]:=35;
    REPEAT G:=G+1;Q1[G]:=Q1[G+1] UNTIL Q1[G+1]=0
   END;
   IF G1=10 THEN
   BEGIN
    Q1[G-1]:=Q1[G];G:=G+1;
    REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
   END;
  UNTIL G1=0;
END;
PROCEDURE FOEND(COLOR:INTEGER);
VAR FOEK:INTEGER;VAR DEGIS:INTEGER;STR:STRING;
BEGIN
 HIDEMOUSE;SETCOLOR(COLOR);SETTEXTSTYLE(2,0,4);
 CASE A[I,3] OF
```

```
 1..7,12..21:OUTTEXTXY(28,6,'OUTPUT=');
 8..11,22,23:BEGIN
  IF (A[I,3]>7)AND(A[I,3]<12) THEN DEGIS:=A[I,20]
  ELSE DEGIS:=A[I,20]-1-CIK;
  INTSTR(DEGIS,STR);
  IF DEGIS>9 THEN FOEK:=0 ELSE FOEK:=7;
  OUTTEXTXY(14+FOEK,6,'Q');OUTTEXTXY(21+FOEK,9,STR);
  OUTTEXTXY(35,6,'(t+1)=');
  IF (A[I,3]>7)AND(A[I,3]<12) THEN
  IF CIKIS=2 THEN OUTTEXTXY(19+FOEK,3,'`');
 END;
END;
G1:=10;G2:=6;G:=0;
REPEAT
 G:=G+1;
 CASE Q1[G] OF
  31:BEGIN OUTTEXTXY(7*G1,G2,'X');G1:=G1+1 END;
  32:BEGIN OUTTEXTXY(7*G1,G2,'Y');G1:=G1+1 END;
  33:BEGIN OUTTEXTXY(7*G1,G2,'Z');G1:=G1+1 END;
  34:BEGIN OUTTEXTXY(7*G1,G2,'W');G1:=G1+1 END;
  35:BEGIN OUTTEXTXY(7*G1,G2,'0');G1:=G1+1 END;
  36:BEGIN OUTTEXTXY(7*G1,G2,'1');G1:=G1+1 END;
  41:BEGIN OUTTEXTXY(7*G1,G2,'.');G1:=G1+1 END;
  43:BEGIN OUTTEXTXY(7*G1,G2,'+');G1:=G1+1 END;
  45:BEGIN
      OUTTEXTXY(7*G1,G2,'+');CIRCLE(2+7*G1,G2+6,3);G1:=G1+1
     END;
  46:BEGIN
      OUTTEXTXY(1+7*G1,G2-2,'.');CIRCLE(2+7*G1,G2+6,3);G1:=G1+1
     END;
  47:BEGIN OUTTEXTXY(7*G1,G2,'`');G1:=G1+1 END;
  51:BEGIN OUTTEXTXY(7*G1,G2,'(');G1:=G1+1 END;
  52:BEGIN OUTTEXTXY(7*G1,G2,')');G1:=G1+1 END;
  101..200:BEGIN
    DEGIS:=Q1[G]-100;INTSTR(DEGIS,STR);
    OUTTEXTXY(7*G1,G2,'Q');OUTTEXTXY(7+7*G1,G2+3,STR);
    IF Q1[G]<110 THEN G1:=G1+2 ELSE G1:=G1+3;
```

```
    END;
   END;
   IF G1>41 THEN BEGIN G1:=1;G2:=G2+12 END;
  UNTIL Q1[G]=0;
  SHOWMOUSE;
 END;
 PROCEDURE INDICATOR;
 VAR HT,HT1,HT2:INTEGER;
 BEGIN
  HT1:=10;HT2:=23;HT:=0;
  REPEAT
   HT:=HT+1;
   CASE Q1[HT] OF
    31..52:HT1:=HT1+1; 101..109:HT1:=HT1+2; 110..200:HT1:=HT1+3;
   END;
   IF HT1>41 THEN IF Q1[HT]<>0 THEN BEGIN HT1:=1;HT2:=HT2+12 END;
  UNTIL Q1[HT]=0;
  IF HT2<94 THEN HT2:=94;
  HIDEMOUSE;
  IF HT2<479 THEN SETVIEWPORT(330,0,639,HT2,TRUE)
  ELSE SETVIEWPORT(330,0,639,479,TRUE);
  CLEARVIEWPORT;SETCOLOR(5);
  RECTANGLE(0,0,309,HT2);RECTANGLE(1,1,308,HT2-1);
  SETCOLOR(15);RECTANGLE(3,3,306,HT2-3);
  SETFILLSTYLE(1,15);FLOODFILL(5,5,15);
  REPEAT
   FOEND(15);DELAY(100);FOEND(0);
   FOR HT:=1 TO 9 DO IF MOUSEBUTTONS=EVNOTHING THEN DELAY(100);
  UNTIL MOUSEBUTTONS<>EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
  REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
  SETVIEWPORT(0,0,639,479,TRUE);SHOWMOUSE;
  DELAY(250);
 END;
 PROCEDURE FIND_OUTPUT;
 LABEL 1,2;
 VAR BAK:INTEGER;
```

```
BEGIN
 MARK_OUTPUT;IF PT=1 THEN EXIT;
 YARDIM:=16;HELP(YARDIM,HELPX);
 HIDEMOUSE;SETVIEWPORT(470,50,490,70,TRUE);
 CLEARVIEWPORT;SETVIEWPORT(0,0,639,479,TRUE);
 WHILE KEYPRESSED DO CH:=UPCASE(READKEY);CH:=' ';
 REPEAT
  SETCOLOR(0);OUTTEXTXY(476,47,CH);
  IF KEYPRESSED THEN CH:=UPCASE(READKEY);
  SETCOLOR(14);OUTTEXTXY(476,47,CH);DELAY(200);
 UNTIL (CH='Y')OR(CH='N');
 SHOWMOUSE;
 IF CH='N' THEN BEGIN YARDIM:=15;HELP(YARDIM,HELPX);PUT;EXIT END;
 YARDIM:=17;HELP(YARDIM,HELPX);
 FOR G:=1 TO SON DO BEGIN Q1[G]:=0;Q2[G]:=0 END;
 FOR G1:=1 TO SON1 DO FOR G2:=1 TO SON2 DO O[G1,G2]:=A[G1,G2];
 FOR G1:=1 TO SON1 DO FOR G2:=1 TO SON3 DO B[G1,G2]:=0;
 FOR G1:=1 TO SON1 DO
 BEGIN
  CASE A[G1,3] OF
   1..7:FOR G:=3 TO 6 DO
        CASE A[G1,2*G] OF
         0,1,3,4,6:O[G1,2*G]:=-1;
         2,5,7:O[G1,2*G]:=-2;
         8:O[G1,2*G]:=-3;
         9:O[G1,2*G]:=-4;
        END;
   8..23:FOR G:=2 TO 7 DO
         CASE A[G1,2*G+1] OF
          0,1,3,4,6:O[G1,2*G+1]:=-1;
          2,5,7:O[G1,2*G+1]:=-2;
          8:O[G1,2*G+1]:=-3;
          9:O[G1,2*G+1]:=-4;
         END;
END END;
FOR HASANT:=1 TO SON1 DO
CASE O[HASANT,3] OF
```

```
 0:B[HASANT,1]:=60;  1..7:FOPART1_7;  8:FOPART8;
 9:FOPART9;  10:FOPART10;  11:FOPART11;  12:FOPART12;
 13:FOPART13;  14:FOPART14;  15:FOPART15;  16:FOPART16;
 17:FOPART17;  18:FOPART18;  19:FOPART19;  20:FOPART20;
 21:FOPART21;  22:FOPART22;  23:FOPART23;
END;
CASE CIKIS OF
 0,1,3,4,6:BEGIN G:=0;CIK:=-1 END;
 2,5,7:BEGIN G:=50;CIK:=-2 END;
 8:BEGIN G:=100;CIK:=-3 END;
 9:BEGIN G:=150;CIK:=-4 END;
END;
IF B[I,G+1]=60 THEN
BEGIN
 YARDIM:=18;HELP(YARDIM,HELPX);
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 YARDIM:=15;HELP(YARDIM,HELPX);PUT;EXIT;
END;
G1:=0;
REPEAT G:=G+1;G1:=G1+1;Q1[G1]:=B[I,G] UNTIL Q1[G1]=0;
1:G:=0;REPEAT G:=G+1;Q2[G]:=Q1[G] UNTIL Q1[G]<31;
IF Q1[G]<>0 THEN
BEGIN
 G1:=G-1;
 CASE Q1[G+1] OF
  -1:G2:=0;   -2:G2:=50;   -3:G2:=100;  -4:G2:=150;
 END;
 REPEAT G1:=G1+1;G2:=G2+1;Q2[G1]:=B[Q1[G],G2] UNTIL Q2[G1]=0;
 G:=G+1;
 REPEAT G:=G+1;Q2[G1]:=Q1[G];G1:=G1+1 UNTIL Q1[G]=0;
 G:=0;
 REPEAT G:=G+1;Q1[G]:=Q2[G] UNTIL Q1[G]=0;GOTO 1;
END;
G:=0;REPEAT G:=G+1 UNTIL (Q1[G]=0)OR(Q1[G]=60);
IF Q1[G]=60 THEN
```

```
BEGIN
 YARDIM:=18;HELP(YARDIM,HELPX);
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
 REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
 YARDIM:=15;HELP(YARDIM,HELPX);PUT;EXIT;
END;
G:=0;
REPEAT
 G:=G+1;Q1[G]:=Q2[G];
 IF Q1[G]=42 THEN Q1[G]:=41;
 IF Q1[G]=44 THEN Q1[G]:=43;
UNTIL Q1[G]=0;
REPEAT
 G:=0;G1:=1;
 REPEAT G:=G+1 UNTIL ((Q1[G]=47)AND(Q1[G+1]=47))OR(Q1[G]=0);
 IF Q1[G]=0 THEN G1:=0
 ELSE REPEAT G:=G+1;Q1[G-1]:=Q1[G+1] UNTIL Q1[G-1]=0;
UNTIL G1=0;
REDUCTION_ZEROS_AND_ONES;
REPEAT
 G:=0;
 REPEAT G:=G+1 UNTIL (Q1[G]=0)OR((Q1[G]=51)AND(Q1[G+2]=52))OR
                     ((Q1[G]=51)AND(Q1[G+2]=47)AND(Q1[G+3]=52))
                     OR((Q1[G]=47)AND(Q1[G+1]=47));
 IF Q1[G]=0 THEN G1:=0 ELSE
 BEGIN
  G1:=1;
  IF (Q1[G]=47)AND(Q1[G+1]=47) THEN
  BEGIN
   G:=G+1;REPEAT G:=G+1;Q1[G-2]:=Q1[G] UNTIL Q1[G]=0
  END ELSE
  BEGIN
   Q1[G]:=Q1[G+1];
   IF Q1[G+2]=47 THEN BEGIN G:=G+1;Q1[G]:=47 END;
   REPEAT G:=G+1;Q1[G]:=Q1[G+2] UNTIL Q1[G+2]=0;
  END;
```

```
   END
  UNTIL G1=0;
  REPEAT
   G:=0;
   2:REPEAT G:=G+1 UNTIL (Q1[G]=0)OR((Q1[G]=51)AND(Q1[G+1]=51));
   IF Q1[G]=0 THEN G1:=0 ELSE
   BEGIN
    G1:=1;BAK:=1;G2:=G+1;
    REPEAT
     G2:=G2+1;IF Q1[G2]=51 THEN BAK:=BAK+1;
     IF Q1[G2]=52 THEN BAK:=BAK-1;
    UNTIL BAK=0;
    IF (Q1[G2+1]=52)OR((Q1[G2+1]=47)AND(Q1[G2+2]=52)) THEN
    BEGIN
     IF (Q1[G2+1]=47)AND(Q1[G2+2]=52) THEN G2:=G2+1;
     REPEAT G:=G+1;Q1[G]:=Q1[G+1] UNTIL G=G2-1;
     REPEAT G:=G+1;Q1[G]:=Q1[G+2] UNTIL Q1[G+2]=0
    END ELSE
    BEGIN
     BAK:=1;
     REPEAT
      G2:=G2+1;IF Q1[G2]=51 THEN BAK:=BAK+1;
      IF Q1[G2]=52 THEN BAK:=BAK-1;
     UNTIL BAK=0;
     G:=G2;GOTO 2;
   END;END;
 UNTIL G1=0;
 INDICATOR;YARDIM:=15;HELP(YARDIM,HELPX);PUT;
END;
PROCEDURE SAVE_OR_TAKE;
LABEL 1;
BEGIN
  HIDEMOUSE;
  1:HELP(23,HELPX);
  IF T=12 THEN OUTTEXTXY(380,15,'DOSYAYA BILGI KAYDETME ISLEMI')
  ELSE OUTTEXTXY(390,15,'DOSYADAN BILGI ALMA ISLEMI');
  OUTTEXTXY(370,40,'DOSYA ADI:');IDOSYA:=0;
```

```
 REPEAT
  REPEAT
   OKEY:=0;CHDOS:=' ';CHDOSYA:=UPCASE(READKEY);
   IF CHDOSYA=#0 THEN CHDOS:=UPCASE(READKEY);
   IF CHDOS=#75 THEN GOTO 1;
   CASE CHDOSYA OF 'A'..'Z',#13:OKEY:=1 END;
  UNTIL OKEY=1;
  IF (CHDOSYA<>#13)AND(IDOSYA<8) THEN
  BEGIN
   IDOSYA:=IDOSYA+1;OUTTEXTXY(430+7*IDOSYA,40,CHDOSYA);
   DOSYA_ISMI[IDOSYA]:=CHDOSYA
  END;
 UNTIL (CHDOSYA=#13)AND(IDOSYA<>0);
 IDOSYA:=IDOSYA+1;DOSYA_ISMI[IDOSYA]:='.';
 OUTTEXTXY(430+7*IDOSYA,40,DOSYA_ISMI[IDOSYA]);
 IDOSYA:=IDOSYA+1;DOSYA_ISMI[IDOSYA]:='H';
 OUTTEXTXY(430+7*IDOSYA,40,DOSYA_ISMI[IDOSYA]);
 IDOSYA:=IDOSYA+1;DOSYA_ISMI[IDOSYA]:='S';
 OUTTEXTXY(430+7*IDOSYA,40,DOSYA_ISMI[IDOSYA]);
 IDOSYA:=IDOSYA+1;DOSYA_ISMI[IDOSYA]:='N';
 OUTTEXTXY(430+7*IDOSYA,40,DOSYA_ISMI[IDOSYA]);
 DOSYA_ISMI:=DOSYA_ISMI[1];FOR JDOSYA:=2 TO IDOSYA DO
 DOSYA_ISMI:=DOSYA_ISMI+DOSYA_ISMI[JDOSYA];
 SHOWMOUSE;ASSIGN(DOSYA,DOSYA_ISMI);
 IF T=12 THEN
 BEGIN
  REWRITE(DOSYA);
  FOR IDOSYA:=1 TO 30 DO FOR JDOSYA:=1 TO 20 DO
  WRITELN(DOSYA,A[IDOSYA,JDOSYA]);
  CLOSE(DOSYA);
 END ELSE
 BEGIN
  {$I-}RESET(DOSYA);{$I+}
  IF IORESULT=0 THEN
  BEGIN
   WHILE NOT(EOF(DOSYA)) DO
   BEGIN
```

```
      FOR IDOSYA:=1 TO 30 DO FOR JDOSYA:=1 TO 20 DO
      READLN(DOSYA,A[IDOSYA,JDOSYA]);
     END;
     CLOSE(DOSYA);PUT;
    END ELSE
    BEGIN
     HELP(23,HELPX);OUTTEXTXY(380,15,'OYLE BIR DOSYA YOK');
     REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
     REPEAT UNTIL MOUSEBUTTONS<>EVNOTHING;
     REPEAT UNTIL MOUSEBUTTONS=EVNOTHING;
    END;
   END;
   RENK0(T,5,CircuitElement);T:=TX;RENK0(T,11,CircuitElement);
   HELP(YARDIM,HELPX);PARCA(CE,T,CircuitElement);
END;
PROCEDURE DIGITAL_LOGIC_CIRCUITS;
BEGIN
 REPEAT
  MOUSE;
  CASE T OF
   1:PUT_CIRCUIT_ELEMENT; 2:TAKE_CIRCUIT_ELEMENT;
   3:MOVE_CIRCUIT_ELEMENT; 4:PUT_WIRE; 5:TAKE_WIRE;
   6:NEW; 7:GOOUT; 8:PUT_INPUT;  9:TAKE_INPUT;
   10:FIND_OUTPUT; 12,13:SAVE_OR_TAKE;
  END;
 UNTIL T=-1;
END;
BEGIN
 GD:=DETECT;INITGRAPH(GD,GM,'');
 IF GRAPHRESULT<>GROK THEN HALT(1);INITEVENTS;
 MUSIC:='SOUND';YERX:=2;YERY:=2;
 CircuitElement:='Put Circuit Element';
 BOXES0(CircuitElement);KONUM(YERX,YERY);HELP(YARDIM,HELPX);
 DIGITAL_LOGIC_CIRCUITS;CLOSEGRAPH;
END.
```

2. TEMURTAS.PAS

```
UNIT TEMURTAS;
{ This unit is used by LOGIC.PAS }

INTERFACE

USES CRT,GRAPH,DRIVERS;

PROCEDURE YUKARIOK(OK1,OK2:INTEGER);
PROCEDURE ASAGIOK(OK1,OK2:INTEGER);
PROCEDURE SOLOK(OK1,OK2:INTEGER);
PROCEDURE SAGOK(OK1,OK2:INTEGER);
PROCEDURE KONUM(YERX,YERY:INTEGER);
PROCEDURE HELP(YARDIM:INTEGER;VAR HELPX:INTEGER);
PROCEDURE RENK0(T,RENK:INTEGER;CircuitElement:STRING);
PROCEDURE BOXES0(CircuitElement:STRING);
PROCEDURE RENK1(CE,RENK:INTEGER);
PROCEDURE BOXES1;
PROCEDURE MARK(AX,AY,MARKX:INTEGER);
PROCEDURE AND_KAPISI(X,Y:INTEGER);
PROCEDURE NAND_KAPISI(X,Y:INTEGER);
PROCEDURE OR_KAPISI(X,Y:INTEGER);
PROCEDURE NOR_KAPISI(X,Y:INTEGER);
PROCEDURE EXOR_KAPISI(X,Y:INTEGER);
PROCEDURE EXNOR_KAPISI(X,Y:INTEGER);
PROCEDURE NOT_KAPISI(X,Y:INTEGER);
PROCEDURE JKFF(X,Y:INTEGER);
PROCEDURE DFF(X,Y:INTEGER);
PROCEDURE TFF(X,Y:INTEGER);
PROCEDURE RSFF(X,Y:INTEGER);
PROCEDURE HALF_ADDER(X,Y:INTEGER);
PROCEDURE FULL_ADDER(X,Y:INTEGER);
PROCEDURE ENCODER21(X,Y:INTEGER);
PROCEDURE ENCODER42(X,Y:INTEGER);
PROCEDURE MULTIPLEXER21(X,Y:INTEGER);
PROCEDURE MULTIPLEXER41(X,Y:INTEGER);
PROCEDURE DEMULTIPLEXER12(X,Y:INTEGER);
PROCEDURE DEMULTIPLEXER14(X,Y:INTEGER);
```

```
 PROCEDURE DECODER12(X,Y:INTEGER);
 PROCEDURE DECODER24(X,Y:INTEGER);
 PROCEDURE REGISTER2(X,Y:INTEGER);
 PROCEDURE REGISTER4(X,Y:INTEGER);
 PROCEDURE NEWPART;
 PROCEDURE GOOUTPART;
 PROCEDURE MOUSEPART(CE:INTEGER;VAR CircuitElement:STRING);
 PROCEDURE CEPART(X,Y,SAYAC:INTEGER;VAR SAY:INTEGER);
 PROCEDURE INTSTR(DEGIS:INTEGER;VAR STR:STRING);
 PROCEDURE PARCA(CE,T:INTEGER;VAR CircuitElement:STRING);

 IMPLEMENTATION

 PROCEDURE YUKARIOK(OK1,OK2:INTEGER);
 BEGIN
  LINE(OK1,OK2+9,OK1,OK2+10);LINE(OK1+1,OK2+8,OK1+1,OK2+10);
  LINE(OK1+2,OK2+6,OK1+2,OK2+10);LINE(OK1+3,OK2+5,OK1+3,OK2+16);
  LINE(OK1+4,OK2+3,OK1+4,OK2+16);LINE(OK1+5,OK2+3,OK1+5,OK2+16);
  LINE(OK1+6,OK2+5,OK1+6,OK2+16);LINE(OK1+7,OK2+6,OK1+7,OK2+10);
  LINE(OK1+8,OK2+8,OK1+8,OK2+10);LINE(OK1+9,OK2+9,OK1+9,OK2+10);
 END;
 PROCEDURE ASAGIOK(OK1,OK2:INTEGER);
 BEGIN
  LINE(OK1,OK2+6,OK1,OK2+7);LINE(OK1+1,OK2+6,OK1+1,OK2+9);
  LINE(OK1+2,OK2+6,OK1+2,OK2+10);LINE(OK1+3,OK2,OK1+3,OK2+11);
  LINE(OK1+4,OK2,OK1+4,OK2+13);LINE(OK1+5,OK2,OK1+5,OK2+13);
  LINE(OK1+6,OK2,OK1+6,OK2+11);LINE(OK1+7,OK2+6,OK1+7,OK2+10);
  LINE(OK1+8,OK2+6,OK1+8,OK2+9);LINE(OK1+9,OK2+6,OK1+9,OK2+7);
 END;
 PROCEDURE SOLOK(OK1,OK2:INTEGER);
 BEGIN
  LINE(OK1+8,OK2,OK1+10,OK2);LINE(OK1+6,OK2+1,OK1+10,OK2+1);
  LINE(OK1+4,OK2+2,OK1+10,OK2+2);LINE(OK1+2,OK2+3,OK1+20,OK2+3);
  LINE(OK1,OK2+4,OK1+20,OK2+4);LINE(OK1,OK2+5,OK1+20,OK2+5);
  LINE(OK1+2,OK2+6,OK1+20,OK2+6);LINE(OK1+4,OK2+7,OK1+10,OK2+7);
  LINE(OK1+6,OK2+8,OK1+10,OK2+8);LINE(OK1+8,OK2+9,OK1+9,OK2+9)
 END;
 PROCEDURE SAGOK(OK1,OK2:INTEGER);
```

```
BEGIN
 LINE(OK1+10,OK2,OK1+12,OK2);LINE(OK1+10,OK2+1,OK1+14,OK2+1);
 LINE(OK1+10,OK2+2,OK1+16,OK2+2);LINE(OK1,OK2+3,OK1+18,OK2+3);
 LINE(OK1,OK2+4,OK1+20,OK2+4);LINE(OK1,OK2+5,OK1+20,OK2+5);
 LINE(OK1,OK2+6,OK1+18,OK2+6);LINE(OK1+10,OK2+7,OK1+16,OK2+7);
 LINE(OK1+10,OK2+8,OK1+14,OK2+8);LINE(OK1+10,OK2+9,OK1+12,OK2+9);
END;
PROCEDURE KONUM(YERX,YERY:INTEGER);
VAR KONUM1,KONUM2:INTEGER;
BEGIN
 HIDEMOUSE;SETVIEWPORT(200,0,329,95,TRUE);
 CLEARVIEWPORT;SETVIEWPORT(0,0,639,479,TRUE);
 SETCOLOR(15);SETFILLSTYLE(1,15);RECTANGLE(200,0,328,94);
 RECTANGLE(208,8,320,88);FLOODFILL(210,10,15);
 SETCOLOR(7);SETFILLSTYLE(1,7);
 RECTANGLE(208+8*YERX,8+8*YERY,288+8*YERX,56+8*YERY);
 FLOODFILL(210+8*YERX,10+8*YERY,7);
 FOR KONUM1:=0 TO 14 DO FOR KONUM2:=0 TO 10 DO
 PUTPIXEL(208+8*KONUM1,8+8*KONUM2,0);
 SHOWMOUSE;
END;
PROCEDURE HELP(YARDIM:INTEGER;VAR HELPX:INTEGER);
BEGIN
 HIDEMOUSE;SETVIEWPORT(330,0,639,94,TRUE);
 CLEARVIEWPORT;SETVIEWPORT(0,0,639,479,TRUE);
 SETCOLOR(15);RECTANGLE(330,0,639,94);SETTEXTSTYLE(2,0,4);
 HELPX:=(HELPX+1) MOD 7;
 CASE HELPX OF
  0:BEGIN
     SETCOLOR(11);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,11);
     FLOODFILL(335,5,11);SETCOLOR(0)
    END;
  1:BEGIN
     SETCOLOR(15);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,15);
     FLOODFILL(335,5,15);SETCOLOR(0)
    END;
  2:BEGIN
```

```
    SETCOLOR(7);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,7);
    FLOODFILL(335,5,7);SETCOLOR(5)
   END;
 3:BEGIN
    SETCOLOR(10);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,10);
    FLOODFILL(335,5,10);SETCOLOR(0)
   END;
 4:BEGIN
    SETCOLOR(14);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,14);
    FLOODFILL(335,5,14);SETCOLOR(5)
   END;
 5:BEGIN
    SETCOLOR(11);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,11);
    FLOODFILL(335,5,11);SETCOLOR(0)
   END;
 6:BEGIN
    SETCOLOR(15);RECTANGLE(332,2,637,92);SETFILLSTYLE(1,15);
    FLOODFILL(335,5,15);SETCOLOR(9)
   END;
END;
CASE YARDIM OF
  0:OUTTEXTXY(340,8,'SAG VEYA SOL MOUSE TUSUNA BASILARAK
    MOUSE CALISIR');
  1:BEGIN
     OUTTEXTXY(340,8,'DEVRE ELEMANI YERLESTIRMEK ICIN MOUSE
     OKU BOS');OUTTEXTXY(340,24,'ALANA GETIRILIP SAG VEYA SOL
     MOUSE TUSUNA BASILIR');
    END;
  2:OUTTEXTXY(340,8,'EN FAZLA 30 DEVRE ELEMANI YERLESTIRILIR');
  3:BEGIN
     OUTTEXTXY(340,8,'MOUSE OKU IPTAL EDILMEK ISTENILEN
     DEVRE ELEMANI');OUTTEXTXY(340,24,'UZERINE GETIRILIP SAG
     VEYA SOL MOUSE TUSUNA');OUTTEXTXY(340,40,'BASILIR');
    END;
  4:BEGIN
     OUTTEXTXY(340,8,'OK ONCE YERI DEGISTIRILMEK ISTENILEN
     DEVRE ELEMANI');OUTTEXTXY(340,24,'UZERINE GETIRILIP MOUSE
```

```
    TUSUNA BASILIR. SONRA OK');OUTTEXTXY(340,40,'BOS ALANA
    GETIRILIP MOUSE TUSUNU BASILIR. DEVRE');
    OUTTEXTXY(340,56,'ELEMANI BIR YERDEN BIR YERE BU
    SEKILDE TASINIR');
   END;
 5:BEGIN
    OUTTEXTXY(340,8,'MOUSE ILE ILK BASILAN DEVRE ELEMANI
    CIKISINDAN');OUTTEXTXY(340,24,'MOUSE ILE IKINCI BASILAN DEVRE
    ELEMANI GIRISINE');OUTTEXTXY(340,40,'HAT DOSER');
   END;
 6:BEGIN
    OUTTEXTXY(340,8,'DEVRE ELEMANLARI ARASINDA DONGUYE IZIN
    VERILMEZ.');OUTTEXTXY(340,24,'NOT: ARADA FLIP_FLOP VEYA
    REGISTER OLURSA HARIC');
   END;
 7:BEGIN
    OUTTEXTXY(340,8,'MOUSE ILE ILK BASILAN DEVRE ELEMANI CIKISI
    ILE');OUTTEXTXY(340,24,'MOUSE ILE IKINCI BASILAN DEVRE ELEMANI
    GIRISI');OUTTEXTXY(340,40,'ARASINDAKI HATI IPTAL EDER');
   END;
 8:OUTTEXTXY(340,8,'ARADA HAT YOK KI IPTAL EDILSIN');
 9:BEGIN
    OUTTEXTXY(340,8,'INPUT GIRMEK ICIN MOUSE OKUNU HERHANGI BIR');
    OUTTEXTXY(340,24,'DEVRE ELEMANI GIRISI UZERINE GETIRIP
    ISARETLE');
   END;
10:BEGIN
    SETCOLOR(0);SETTEXTSTYLE(1,0,2);
    OUTTEXTXY(400,15,'X  Y  Z  W  0  1');
    SETTEXTSTYLE(3,0,1);
    OUTTEXTXY(380,45,'Karekterlerinden birini gir');
   END;
11:BEGIN
    OUTTEXTXY(340,8,'INPUT IPTAL ETMEK ICIN MOUSE OKUNU HERHANGI
    BIR');OUTTEXTXY(340,24,'DEVRE ELEMANI GIRISI UZERINE GETIRIP
    ISARETLE');
   END;
```

```
12:BEGIN
    OUTTEXTXY(340,8,'INPUT YOK VEYA SADECE DIGER DEVRE');
    OUTTEXTXY(340,24,'ELEMANLARINDAN GELEN INPUT VEYA
    INPUTLAR VAR');
   END;
13:BEGIN
    SETCOLOR(0);SETTEXTSTYLE(1,0,2);
    OUTTEXTXY(400,15,'X  Y  Z  W  0  1');
    SETTEXTSTYLE(3,0,1);
    OUTTEXTXY(360,45,'Karekterlerinden birini geri al');
   END;
14:OUTTEXTXY(340,8,'GIRILEN INPUT YOK KI IPTAL EDILSIN');
15:BEGIN
    OUTTEXTXY(340,8,'OK HESAPLANILMAK ISTENILEN DEVRE ELEMANI
    CIKTISI');OUTTEXTXY(336,24,'UZERINE GETIRILIP SAG VEYA SOL
    MOUSE TUSUNU BASILIR');
   END;
16:BEGIN
    SETCOLOR(0);SETTEXTSTYLE(3,0,1);
    OUTTEXTXY(338,20,'HESAPLAMAK ISTIYOR MUSUN?Y/N');
   END;
17:BEGIN
    SETCOLOR(0);SETTEXTSTYLE(1,0,3);
    OUTTEXTXY(395,30,'HESAPLANIYOR');DELAY(300)
   END;
18:OUTTEXTXY(340,8,'GIRDILER EKSIK. GIRDILERI TAMAMLA');
19:BEGIN
    OUTTEXTXY(340,8,'NOT KAPISI EN FAZLA 1 GIRDI KABUL EDER.
    DIGER ');OUTTEXTXY(340,24,'KAPILAR ISE EN FAZLA 4 GIRDI
    KABUL EDER');
   END;
20:OUTTEXTXY(340,8,'DEVRE ELEMANININ O GIRISI BOS DEGIL');
21:BEGIN
    OUTTEXTXY(340,8,'DEVRE ELEMANININ O GIRISI DOLU DEGIL');
    OUTTEXTXY(340,24,'VEYA DIGER BIR DEVRE ELEMANININ
    OUTPUTU GIRMIS');
   END;
```

```
  22:OUTTEXTXY(340,8,'DEVRE ELEMANININ O GIRISINDE GIRILEN
     INPUT YOK');
  23:BEGIN END;
 END;
 SHOWMOUSE
END;
PROCEDURE RENK0(T,RENK:INTEGER;CircuitElement:STRING);
VAR OK:INTEGER;
BEGIN
 SETCOLOR(RENK);SETTEXTSTYLE(2,0,4);HIDEMOUSE;
 CASE T OF
  1:BEGIN
     SETVIEWPORT(0,0,158,22,TRUE);CLEARVIEWPORT;
     SETVIEWPORT(0,0,639,479,TRUE);SETFILLSTYLE(1,7);
     SETCOLOR(15);RECTANGLE(0,0,158,22);
     SETCOLOR(8);RECTANGLE(2,2,156,20);FLOODFILL(5,5,8);
     SETCOLOR(RENK);OUTTEXTXY(25,5,CircuitElement);
    END;
  2:OUTTEXTXY(5,29,'TakeE');  3:OUTTEXTXY(5,53,'MoveE');
  4:OUTTEXTXY(8,77,'PutW');   5:OUTTEXTXY(45,29,'TakeW');
  6:OUTTEXTXY(50,53,'NEW');   7:OUTTEXTXY(48,77,'EXIT');
  8:OUTTEXTXY(90,29,'PutI');  9:OUTTEXTXY(85,53,'TakeI');
  10:OUTTEXTXY(85,77,'FindO'); 11:OUTTEXTXY(130,29,'SES');
  12:OUTTEXTXY(127,53,'SAVE'); 13:OUTTEXTXY(126,77,'FILES');
  14:YUKARIOK(174,2);  15:ASAGIOK(174,29);
  16:SAGOK(169,55);  17:SOLOK(169,79);
 END;
 SHOWMOUSE;
END;
PROCEDURE BOXES0(CircuitElement:STRING);
VAR BOX1,BOX2,OK:INTEGER;
BEGIN
 HIDEMOUSE;SETVIEWPORT(0,0,199,95,TRUE);CLEARVIEWPORT;
 SETVIEWPORT(0,0,639,479,TRUE);SETFILLSTYLE(1,7);
 SETCOLOR(15);RECTANGLE(0,0,158,22);RECTANGLE(160,0,198,22);
 SETCOLOR(8);RECTANGLE(2,2,156,20);RECTANGLE(162,2,196,20);
 FLOODFILL(5,5,8);FLOODFILL(165,5,8);
```

```
 FOR BOX1:=0 TO 4 DO FOR BOX2:=1 TO 3 DO
 BEGIN
  SETCOLOR(15);RECTANGLE(40*BOX1,24*BOX2,40*BOX1+38,24*BOX2+22);
  SETCOLOR(8);RECTANGLE(40*BOX1+2,24*BOX2+2,40*BOX1+36,24*BOX2+20);
  FLOODFILL(40*BOX1+5,24*BOX2+5,8);
 END;
 SETCOLOR(5);SETTEXTSTYLE(2,0,4);
 OUTTEXTXY(25,5,CircuitElement);
 OUTTEXTXY(5,29,'TakeE');OUTTEXTXY(5,53,'MoveE');
 OUTTEXTXY(8,77,'PutW');OUTTEXTXY(45,29,'TakeW');
 OUTTEXTXY(50,53,'NEW');OUTTEXTXY(48,77,'EXIT');
 OUTTEXTXY(90,29,'PutI');OUTTEXTXY(85,53,'TakeI');
 OUTTEXTXY(85,77,'FindO');OUTTEXTXY(130,29,'SES');
 OUTTEXTXY(127,53,'SAVE');OUTTEXTXY(126,77,'FILES');
 YUKARIOK(174,2);ASAGIOK(174,29);SAGOK(169,55);
 SOLOK(169,79);SHOWMOUSE;
END;
PROCEDURE RENK1(CE,RENK:INTEGER);
BEGIN
 SETCOLOR(RENK);SETTEXTSTYLE(2,0,4);HIDEMOUSE;
 CASE CE OF
  1:OUTTEXTXY(11,5,'AND Kap□s□');2:OUTTEXTXY(89,5,'NAND Kap□s□');
  3:OUTTEXTXY(173,5,'OR Kap□s□');4:OUTTEXTXY(250,5,'NOR Kap□s□');
  5:OUTTEXTXY(328,5,'EXOR Kap□s□');6:OUTTEXTXY(405,5,'EXNOR Kap□s□');
  7:OUTTEXTXY(491,5,'NOT Kap□s□');8:OUTTEXTXY(564,5,'JK Flib-Flob');
  9:OUTTEXTXY(7,29,'D Flip-Flop');10:OUTTEXTXY(86,29,'T Flip-Flop');
  11:OUTTEXTXY(165,29,'RS Flip-Flop');
  12:OUTTEXTXY(251,29,'Half Adder');
  13:OUTTEXTXY(330,29,'Full Adder');
  14:OUTTEXTXY(406,29,'2*1 Encoder');
  15:OUTTEXTXY(486,29,'4*2 Encoder');
  16:OUTTEXTXY(576,29,'2*1 MUX.');
  17:OUTTEXTXY(16,53,'4*1 MUX.');
  18:OUTTEXTXY(90,53,'1*2 DEMUX.');
  19:OUTTEXTXY(170,53,'1*4 DEMUX.');
  20:OUTTEXTXY(248,53,'1*2 Decoder');
  21:OUTTEXTXY(328,53,'2*4 Decoder');
```

```
  22:OUTTEXTXY(408,53,'2bitRegister');
  23:OUTTEXTXY(486,53,'4bitRegister');
  24:OUTTEXTXY(564,53,'Empty');25:OUTTEXTXY(11,77,'Empty');
  26:OUTTEXTXY(89,77,'Empty');27:OUTTEXTXY(173,77,'Empty');
  28:OUTTEXTXY(250,77,'Empty');29:OUTTEXTXY(328,77,'Empty');
  30:OUTTEXTXY(405,77,'Empty');31:OUTTEXTXY(491,77,'Empty');
  32:OUTTEXTXY(564,77,'Empty');
 END;
 SHOWMOUSE;
END;
PROCEDURE BOXES1;
VAR BOX1,BOX2:INTEGER;
BEGIN
 HIDEMOUSE;SETVIEWPORT(0,0,639,95,TRUE);CLEARVIEWPORT;
 SETVIEWPORT(0,0,639,479,TRUE);SETFILLSTYLE(1,7);
 FOR BOX1:=0 TO 7 DO FOR BOX2:=0 TO 3 DO
 BEGIN
  SETCOLOR(15);RECTANGLE(80*BOX1,24*BOX2,80*BOX1+78,24*BOX2+22);
  SETCOLOR(8);RECTANGLE(80*BOX1+2,24*BOX2+2,80*BOX1+76,24*BOX2+20);
  FLOODFILL(80*BOX1+5,24*BOX2+5,8);
 END;
 SETCOLOR(5);SETTEXTSTYLE(2,0,4);
 OUTTEXTXY(11,5,'AND Kap□s□');OUTTEXTXY(89,5,'NAND Kap□s□');
 OUTTEXTXY(173,5,'OR Kap□s□');OUTTEXTXY(250,5,'NOR Kap□s□');
 OUTTEXTXY(328,5,'EXOR Kap□s□');OUTTEXTXY(405,5,'EXNOR Kap□s□');
 OUTTEXTXY(491,5,'NOT Kap□s□');OUTTEXTXY(564,5,'JK Flip-Flop');
 OUTTEXTXY(7,29,'D Flip-Flop');OUTTEXTXY(86,29,'T Flip-Flop');
 OUTTEXTXY(165,29,'RS Flip-Flop');OUTTEXTXY(251,29,'Half Adder');
 OUTTEXTXY(330,29,'Full Adder');OUTTEXTXY(406,29,'2*1 Encoder');
 OUTTEXTXY(486,29,'4*2 Encoder');OUTTEXTXY(576,29,'2*1 MUX.');
 OUTTEXTXY(16,53,'4*1 MUX.');OUTTEXTXY(90,53,'1*2 DEMUX.');
 OUTTEXTXY(170,53,'1*4 DEMUX.');OUTTEXTXY(248,53,'1*2 Decoder');
 OUTTEXTXY(328,53,'2*4 Decoder');OUTTEXTXY(408,53,'2bitRegister');
 OUTTEXTXY(486,53,'4bitRegister');OUTTEXTXY(564,53,'Empty');
 OUTTEXTXY(11,77,'Empty');OUTTEXTXY(89,77,'Empty');
 OUTTEXTXY(173,77,'Empty');OUTTEXTXY(250,77,'Empty');
 OUTTEXTXY(328,77,'Empty');OUTTEXTXY(405,77,'Empty');
```

```
 OUTTEXTXY(491,77,'Empty');OUTTEXTXY(564,77,'Empty');
 SHOWMOUSE;
END;
PROCEDURE MARK(AX,AY,MARKX:INTEGER);
BEGIN
 SETCOLOR(10);HIDEMOUSE;
 CASE MARKX OF
  1:BEGIN
     LINE(AX+12,AY-4,AX+22,AY+6);LINE(AX+22,AY-4,AX+12,AY+6)
    END;
  2:BEGIN
     LINE(AX+6,AY-14,AX+14,AY-6);LINE(AX+14,AY-14,AX+6,AY-6)
    END;
   3:BEGIN LINE(AX+6,AY-4,AX+14,AY+4);LINE(AX+14,AY-4,AX+6,AY+4) END;
   4:BEGIN
      LINE(AX+6,AY+6,AX+14,AY+14);LINE(AX+14,AY+6,AX+6,AY+14)
     END;
   5:BEGIN LINE(AX,AY-24,AX+8,AY-16);LINE(AX,AY-16,AX+8,AY-24) END;
   6:BEGIN LINE(AX,AY-16,AX+8,AY-8);LINE(AX,AY-8,AX+8,AY-16) END;
   7:BEGIN LINE(AX,AY-8,AX+8,AY);LINE(AX,AY,AX+8,AY-8) END;
   8:BEGIN LINE(AX,AY,AX+8,AY+8);LINE(AX,AY+8,AX+8,AY) END;
   9:BEGIN LINE(AX,AY+16,AX+8,AY+24);LINE(AX,AY+24,AX+8,AY+16) END;
  10:BEGIN LINE(AX,AY+24,AX+8,AY+32);LINE(AX,AY+32,AX+8,AY+24) END;
  11:BEGIN LINE(AX,AY-25,AX+8,AY-17);LINE(AX,AY-17,AX+8,AY-25) END;
  12:BEGIN LINE(AX,AY-11,AX+8,AY-3);LINE(AX,AY-3,AX+8,AY-11) END;
  13:BEGIN LINE(AX,AY+3,AX+8,AY+11);LINE(AX,AY+11,AX+8,AY+3) END;
  14:BEGIN LINE(AX,AY+17,AX+8,AY+25);LINE(AX,AY+25,AX+8,AY+17) END;
  15:BEGIN LINE(AX,AY-22,AX+8,AY-14);LINE(AX,AY-14,AX+8,AY-22) END;
  16:BEGIN LINE(AX,AY-4,AX+8,AY+4);LINE(AX,AY+5,AX+8,AY-5) END;
  17:BEGIN LINE(AX,AY+14,AX+8,AY+22);LINE(AX,AY+22,AX+8,AY+14) END;
  18:BEGIN LINE(AX,AY-19,AX+8,AY-11);LINE(AX,AY-11,AX+8,AY-19) END;
  19:BEGIN LINE(AX,AY+19,AX+8,AY+11);LINE(AX,AY+11,AX+8,AY+19) END;
  20:BEGIN
      LINE(AX+27,AY-14,AX+35,AY-6);LINE(AX+35,AY-14,AX+27,AY-6)
     END;
  21:BEGIN
      LINE(AX+27,AY-4,AX+35,AY+4);LINE(AX+35,AY-4,AX+27,AY+4)
```

```
      END;
   22:BEGIN
       LINE(AX+27,AY+6,AX+35,AY+14);LINE(AX+35,AY+6,AX+27,AY+14)
      END;
   23:BEGIN
       LINE(AX+34,AY-25,AX+42,AY-17);LINE(AX+34,AY-17,AX+42,AY-25)
      END;
   24:BEGIN
       LINE(AX+34,AY-11,AX+42,AY-3);LINE(AX+34,AY-3,AX+42,AY-11)
      END;
   25:BEGIN
       LINE(AX+34,AY+3,AX+42,AY+11);LINE(AX+34,AY+11,AX+42,AY+3)
      END;
   26:BEGIN
       LINE(AX+34,AY+17,AX+42,AY+25);LINE(AX+34,AY+25,AX+42,AY+17)
      END;
   27:BEGIN
       LINE(AX+34,AY-19,AX+42,AY-11);LINE(AX+34,AY-11,AX+42,AY-19)
      END;
   28:BEGIN
       LINE(AX+34,AY+19,AX+42,AY+11);LINE(AX+34,AY+11,AX+42,AY+19)
      END;
   29:BEGIN
       LINE(AX+34,AY+4,AX+42,AY-4);LINE(AX+34,AY-4,AX+42,AY+4)
      END;
   30:BEGIN
       LINE(AX+17,AY-4,AX+27,AY+6);LINE(AX+27,AY-4,AX+17,AY+6)
      END;
 END;
 SHOWMOUSE;
END;
PROCEDURE AND_KAPISI(X,Y:INTEGER);
BEGIN
 ELLIPSE(X,Y,270,90,36,18);LINE(X,Y-18,X,Y+18);
 LINE(X+37,Y,X+44,Y);FLOODFILL(X+13,Y,7)
END;
PROCEDURE NAND_KAPISI(X,Y:INTEGER);
```

```
 BEGIN
  ELLIPSE(X,Y,270,90,32,18);LINE(X,Y-18,X,Y+18);
  CIRCLE(X+35,Y,3);LINE(X+39,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE OR_KAPISI(X,Y:INTEGER);
 BEGIN
  ELLIPSE(X,Y,270,90,36,18);ELLIPSE(X,Y,270,90,7,18);
  LINE(X+37,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE NOR_KAPISI(X,Y:INTEGER);
 BEGIN
  ELLIPSE(X,Y,270,90,32,18);ELLIPSE(X,Y,270,90,7,18);
  CIRCLE(X+35,Y,3);LINE(X+39,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE EXOR_KAPISI(X,Y:INTEGER);
 BEGIN
  ELLIPSE(X,Y,270,90,36,18);ELLIPSE(X,Y,270,90,7,18);
  ELLIPSE(X+3,Y,270,90,7,18);LINE(X+37,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE EXNOR_KAPISI(X,Y:INTEGER);
 BEGIN
  ELLIPSE(X,Y,270,90,32,18);ELLIPSE(X,Y,270,90,7,18);
  ELLIPSE(X+3,Y,270,90,7,18);CIRCLE(X+35,Y,3);
  LINE(X+39,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE NOT_KAPISI(X,Y:INTEGER);
 BEGIN
  LINE(X-6,Y,X+4,Y);LINE(X+5,Y-10,X+5,Y+10);
  LINE(X+5,Y+10,X+27,Y);LINE(X+5,Y-10,X+27,Y);
  CIRCLE(X+30,Y,3);LINE(X+33,Y,X+44,Y);FLOODFILL(X+13,Y,7)
 END;
 PROCEDURE JKFF(X,Y:INTEGER);
 BEGIN
  RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X-6,Y-10,X+3,Y-10);
  LINE(X,Y,X+3,Y);OUTTEXTXY(X-12,Y-6,'CP');
  LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
  LINE(X+4,Y-5,X+9,Y);LINE(X+4,Y+5,X+9,Y);FLOODFILL(X+13,Y,7);
```

```
 OUTTEXTXY(X+30,Y-16,'Q');OUTTEXTXY(X+30,Y+4,'Q');
 OUTTEXTXY(X+30,Y-1,'-');LINE(X-6,Y+10,X+3,Y+10);
 OUTTEXTXY(X+7,Y-16,'J');OUTTEXTXY(X+7,Y+4,'K')
END;
PROCEDURE DFF(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X-6,Y-10,X+3,Y-10);
 LINE(X,Y,X+3,Y);OUTTEXTXY(X-12,Y-6,'CP');
 LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
 LINE(X+4,Y-5,X+9,Y);LINE(X+4,Y+5,X+9,Y);FLOODFILL(X+13,Y,7);
 OUTTEXTXY(X+30,Y-16,'Q');OUTTEXTXY(X+30,Y+4,'Q');
 OUTTEXTXY(X+30,Y-1,'-');OUTTEXTXY(X+7,Y-16,'D');
END;
PROCEDURE TFF(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X-6,Y-10,X+3,Y-10);
 LINE(X,Y,X+3,Y);OUTTEXTXY(X-12,Y-6,'CP');
 LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
 LINE(X+4,Y-5,X+9,Y);LINE(X+4,Y+5,X+9,Y);FLOODFILL(X+13,Y,7);
 OUTTEXTXY(X+30,Y-16,'Q');OUTTEXTXY(X+30,Y+4,'Q');
 OUTTEXTXY(X+30,Y-1,'-');OUTTEXTXY(X+7,Y-16,'T');
END;
PROCEDURE RSFF(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X-6,Y-10,X+3,Y-10);
 LINE(X,Y,X+3,Y);OUTTEXTXY(X-12,Y-6,'CP');
 LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
 LINE(X+4,Y-5,X+9,Y);LINE(X+4,Y+5,X+9,Y);FLOODFILL(X+13,Y,7);
 OUTTEXTXY(X+30,Y-16,'Q');OUTTEXTXY(X+30,Y+4,'Q');
 OUTTEXTXY(X+30,Y-1,'-');LINE(X-6,Y+10,X+3,Y+10);
 OUTTEXTXY(X+7,Y-16,'S');OUTTEXTXY(X+7,Y+4,'R')
END;
PROCEDURE HALF_ADDER(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X-6,Y-10,X+3,Y-10);
 LINE(X-6,Y+10,X+3,Y+10);LINE(X+38,Y-10,X+44,Y-10);
 LINE(X+38,Y+10,X+44,Y+10);FLOODFILL(X+13,Y,7);
```

```
 OUTTEXTXY(X+30,Y-16,'S');OUTTEXTXY(X+30,Y+4,'C');
 OUTTEXTXY(X+7,Y-16,'I1');OUTTEXTXY(X+7,Y+4,'I2');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-8,'H.A');
END;
PROCEDURE FULL_ADDER(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-18,X-4,Y-18);LINE(X-13,Y,X-4,Y);
 LINE(X-13,Y+18,X-4,Y+18);LINE(X+45,Y-15,X+51,Y-15);
 LINE(X+45,Y+15,X+51,Y+15);OUTTEXTXY(X+2,Y-24,'I1');
 OUTTEXTXY(X+2,Y-6,'I2');OUTTEXTXY(X+2,Y+12,'I3');
 OUTTEXTXY(X+36,Y-21,'S');OUTTEXTXY(X+36,Y+9,'C');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-8,'F.A');
END;
PROCEDURE ENCODER21(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);FLOODFILL(X+13,Y,7);
 LINE(X-6,Y-10,X+3,Y-10);LINE(X-6,Y+10,X+3,Y+10);
 LINE(X+38,Y,X+44,Y);OUTTEXTXY(X+7,Y-16,'I0');
 OUTTEXTXY(X+7,Y+4,'I1');OUTTEXTXY(X+30,Y-6,'F');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+16,Y-14,'Enc.');
END;
PROCEDURE ENCODER42(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-21,X-4,Y-21);LINE(X-13,Y-7,X-4,Y-7);
 LINE(X-13,Y+7,X-4,Y+7);LINE(X-13,Y+21,X-4,Y+21);
 LINE(X+45,Y-15,X+51,Y-15);LINE(X+45,Y+15,X+51,Y+15);
 OUTTEXTXY(X+2,Y-27,'I0');OUTTEXTXY(X+2,Y-13,'I1');
 OUTTEXTXY(X+2,Y+1,'I2');OUTTEXTXY(X+2,Y+15,'I3');
 OUTTEXTXY(X+32,Y-21,'F0');OUTTEXTXY(X+32,Y+9,'F1');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+15,Y-22,'Encoder');
END;
PROCEDURE MULTIPLEXER21(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-18,X-4,Y-18);LINE(X-13,Y,X-4,Y);
```

```
 LINE(X-13,Y+18,X-4,Y+18);LINE(X+45,Y,X+51,Y);
 OUTTEXTXY(X+2,Y-24,'I0');OUTTEXTXY(X+2,Y-6,'I1');
 OUTTEXTXY(X+2,Y+12,'S');OUTTEXTXY(X+36,Y-6,'F');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-10,'Mux.');
END;
PROCEDURE MULTIPLEXER41(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-20,X-4,Y-20);LINE(X-13,Y-12,X-4,Y-12);
 LINE(X-13,Y-4,X-4,Y-4);LINE(X-13,Y+4,X-4,Y+4);
 LINE(X-13,Y+20,X-4,Y+20);LINE(X-13,Y+28,X-4,Y+28);
 LINE(X+45,Y,X+51,Y);OUTTEXTXY(X+2,Y-26,'I0');
 OUTTEXTXY(X+2,Y-18,'I1');OUTTEXTXY(X+2,Y-10,'I2');
 OUTTEXTXY(X+2,Y-2,'I3');OUTTEXTXY(X+2,Y+12,'S1');
 OUTTEXTXY(X+2,Y+22,'S0');OUTTEXTXY(X+37,Y-6,'F');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-32,'Muxtiplexer');
END;
PROCEDURE DEMULTIPLEXER12(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);FLOODFILL(X+13,Y,7);
 LINE(X-6,Y-10,X+3,Y-10);LINE(X-6,Y+10,X+3,Y+10);
 LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
 OUTTEXTXY(X+7,Y-16,'I');OUTTEXTXY(X+7,Y+4,'S');
 OUTTEXTXY(X+30,Y-16,'0');OUTTEXTXY(X+30,Y+4,'1');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+15,Y-19,'Demux.');
END;
PROCEDURE DEMULTIPLEXER14(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-18,X-4,Y-18);LINE(X-13,Y,X-4,Y);
 LINE(X-13,Y+18,X-4,Y+18);LINE(X+45,Y-21,X+51,Y-21);
 LINE(X+45,Y-7,X+51,Y-7);LINE(X+45,Y+7,X+51,Y+7);
 LINE(X+45,Y+21,X+51,Y+21);OUTTEXTXY(X+2,Y-24,'I');
 OUTTEXTXY(X+2,Y-6,'S1');OUTTEXTXY(X+2,Y+12,'S0');
 OUTTEXTXY(X+36,Y-27,'0');OUTTEXTXY(X+36,Y-13,'1');
 OUTTEXTXY(X+36,Y+1,'2');OUTTEXTXY(X+36,Y+15,'3');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-19,'Demux.');
```

```
END;
PROCEDURE DECODER12(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);FLOODFILL(X+13,Y,7);
 LINE(X-6,Y,X+3,Y);
 LINE(X+38,Y-10,X+44,Y-10);LINE(X+38,Y+10,X+44,Y+10);
 OUTTEXTXY(X+7,Y-6,'I');
 OUTTEXTXY(X+30,Y-16,'0');OUTTEXTXY(X+30,Y+4,'1');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+15,Y-19,'Decod.');
END;
PROCEDURE DECODER24(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);FLOODFILL(X,Y,7);
 LINE(X-13,Y-15,X-4,Y-15);LINE(X-13,Y+15,X-4,Y+15);
 LINE(X+45,Y-21,X+51,Y-21);LINE(X+45,Y-7,X+51,Y-7);
 LINE(X+45,Y+7,X+51,Y+7);LINE(X+45,Y+21,X+51,Y+21);
 OUTTEXTXY(X+2,Y-21,'I0');OUTTEXTXY(X+2,Y+11,'I1');
 OUTTEXTXY(X+36,Y-27,'0');OUTTEXTXY(X+36,Y-13,'1');
 OUTTEXTXY(X+36,Y+1,'2');OUTTEXTXY(X+36,Y+15,'3');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-22,'Decoder');
END;
PROCEDURE REGISTER2(X,Y:INTEGER);
BEGIN
 RECTANGLE(X+4,Y-18,X+37,Y+18);LINE(X+15,Y+18,X+20,Y+12);
 LINE(X+25,Y+18,X+20,Y+12);LINE(X+20,Y+19,X+20,Y+23);
 OUTTEXTXY(X+15,Y+20,'CP');LINE(X-6,Y-10,X+3,Y-10);
 LINE(X-6,Y+10,X+3,Y+10);LINE(X+38,Y-10,X+44,Y-10);
 LINE(X+38,Y+10,X+44,Y+10);FLOODFILL(X+13,Y,7);
 OUTTEXTXY(X+7,Y-16,'R0');OUTTEXTXY(X+7,Y+4,'R1');
 OUTTEXTXY(X+30,Y-16,'0');OUTTEXTXY(X+30,Y+4,'1');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+17,Y-12,'Reg.');
END;
PROCEDURE REGISTER4(X,Y:INTEGER);
BEGIN
 RECTANGLE(X-3,Y-33,X+44,Y+33);LINE(X+16,Y+33,X+21,Y+27);
 LINE(X+26,Y+33,X+21,Y+27);FLOODFILL(X,Y,7);
 OUTTEXTXY(X+16,Y+37,'CP');LINE(X+21,Y+34,X+21,Y+39);
```

```
 LINE(X-13,Y-21,X-4,Y-21);LINE(X-13,Y-7,X-4,Y-7);
 LINE(X-13,Y+7,X-4,Y+7);LINE(X-13,Y+21,X-4,Y+21);
 LINE(X+45,Y-21,X+51,Y-21);LINE(X+45,Y-7,X+51,Y-7);
 LINE(X+45,Y+7,X+51,Y+7);LINE(X+45,Y+21,X+51,Y+21);
 OUTTEXTXY(X+2,Y-27,'R0');OUTTEXTXY(X+2,Y-13,'R1');
 OUTTEXTXY(X+2,Y+1,'R2');OUTTEXTXY(X+2,Y+15,'R3');
 OUTTEXTXY(X+36,Y-27,'0');OUTTEXTXY(X+36,Y-13,'1');
 OUTTEXTXY(X+36,Y+1,'2');OUTTEXTXY(X+36,Y+15,'3');
 SETTEXTSTYLE(2,1,4);OUTTEXTXY(X+16,Y-26,'Register');
END;
PROCEDURE NEWPART;
BEGIN
 SETVIEWPORT(330,0,639,94,TRUE);
 CLEARVIEWPORT;SETVIEWPORT(0,0,639,479,TRUE);
 SETCOLOR(15);RECTANGLE(330,0,639,94);SETTEXTSTYLE(2,0,4);
 SETCOLOR(11);RECTANGLE(332,2,637,92);
 RECTANGLE(519,10,541,37);SETFILLSTYLE(1,11);
 FLOODFILL(335,5,11);SETTEXTSTYLE(1,0,2);SETCOLOR(0);
 OUTTEXTXY(367,10,'NEW (Y/N) ? :');
END;
PROCEDURE GOOUTPART;
BEGIN
 SETVIEWPORT(330,0,639,94,TRUE);
 CLEARVIEWPORT;SETVIEWPORT(0,0,639,479,TRUE);
 SETCOLOR(15);RECTANGLE(330,0,639,94);SETTEXTSTYLE(2,0,4);
 SETCOLOR(10);RECTANGLE(332,2,637,92);RECTANGLE(602,10,624,37);
 SETFILLSTYLE(1,10);FLOODFILL(335,5,10);
 SETTEXTSTYLE(1,0,2);SETCOLOR(0);
 OUTTEXTXY(352,10,'ARE YOU SURE (Y/N) ? :');
END;
PROCEDURE MOUSEPART(CE:INTEGER;VAR CircuitElement:STRING);
BEGIN
  CASE CE OF
    1:CircuitElement:='   AND GATE';
    2:CircuitElement:='   NAND GATE';
    3:CircuitElement:='    OR GATE';
    4:CircuitElement:='   NOR GATE';
```

```
    5:CircuitElement:='   EXOR GATE';
    6:CircuitElement:='   EXNOR GATE';
    7:CircuitElement:='    NOT GATE';
    8:CircuitElement:='  JK FLIP-FLOP';
    9:CircuitElement:='   D FLOP-FLOP';
   10:CircuitElement:='   T FLIP-FLOP';
   11:CircuitElement:='   RS FLIP-FLOP';
   12:CircuitElement:='    HALF ADDER';
   13:CircuitElement:='    FULL ADDER';
   14:CircuitElement:='   2*1 ENCODER';
   15:CircuitElement:='   4*2 ENCODER';
   16:CircuitElement:='    2*1 MUX.';
   17:CircuitElement:='    4*1 MUX.';
   18:CircuitElement:='   1*2 DEMUX.';
   19:CircuitElement:='   1*4 DEMUX.';
   20:CircuitElement:='  1*2 DECODER';
   21:CircuitElement:='  2*4 DECODER';
   22:CircuitElement:='  2 Bit REGISTER';
   23:CircuitElement:='  4 Bit REGISTER';
  END;
END;
PROCEDURE CEPART(X,Y,SAYAC:INTEGER;VAR SAY:INTEGER);
VAR SAYAR:INTEGER;ST:STRING;
BEGIN
 FOR SAYAR:=1 TO SAYAC DO
 BEGIN
  SAY:=SAY+1;
  CASE SAY OF
   1:ST:='Q1';2:ST:='Q2';3:ST:='Q3';4:ST:='Q4';5:ST:='Q5';
   6:ST:='Q6';7:ST:='Q7';8:ST:='Q8';9:ST:='Q9';10:ST:='Q10';
   11:ST:='Q11';12:ST:='Q12';13:ST:='Q13';14:ST:='Q14';
   15:ST:='Q15';16:ST:='Q16';17:ST:='Q17';18:ST:='Q18';
   19:ST:='Q19';20:ST:='Q20';21:ST:='Q21';22:ST:='Q22';
   23:ST:='Q23';24:ST:='Q24';25:ST:='Q25';26:ST:='Q26';
   27:ST:='Q27';28:ST:='Q28';29:ST:='Q29';30:ST:='Q30';
   31:ST:='Q31';32:ST:='Q32';33:ST:='Q33';34:ST:='Q34';
   35:ST:='Q35';36:ST:='Q36';37:ST:='Q37';38:ST:='Q38';
```

```
    39:ST:='Q39';40:ST:='Q40';41:ST:='Q41';42:ST:='Q42';
    43:ST:='Q43';44:ST:='Q44';45:ST:='Q45';46:ST:='Q46';
    47:ST:='Q47';48:ST:='Q48';49:ST:='Q49';50:ST:='Q50';
    51:ST:='Q51';52:ST:='Q52';53:ST:='Q53';54:ST:='Q54';
    55:ST:='Q55';56:ST:='Q56';57:ST:='Q57';58:ST:='Q58';
    59:ST:='Q59';60:ST:='Q60';61:ST:='Q61';62:ST:='Q62';
    63:ST:='Q63';64:ST:='Q64';65:ST:='Q65';66:ST:='Q66';
    67:ST:='Q67';68:ST:='Q68';69:ST:='Q69';70:ST:='Q70';
    71:ST:='Q71';72:ST:='Q72';73:ST:='Q73';74:ST:='Q74';
    75:ST:='Q75';76:ST:='Q76';77:ST:='Q77';78:ST:='Q78';
    79:ST:='Q79';80:ST:='Q80';81:ST:='Q81';82:ST:='Q82';
    83:ST:='Q83';84:ST:='Q84';85:ST:='Q85';86:ST:='Q86';
    87:ST:='Q87';88:ST:='Q88';89:ST:='Q89';90:ST:='Q90';
    91:ST:='Q91';92:ST:='Q92';93:ST:='Q93';94:ST:='Q94';
    95:ST:='Q95';96:ST:='Q96';97:ST:='Q97';98:ST:='Q98';
    99:ST:='Q99';100:ST:='Q100';
   END;
   SETTEXTSTYLE(2,0,4);
   IF (SAYAC=1)OR((SAYAC=2)AND(SAYAR=1)) THEN
   OUTTEXTXY(X+40,Y-22,ST);
   IF (SAYAC=2)AND(SAYAR=2) THEN OUTTEXTXY(X+40,Y-2,ST);
   IF (SAYAC=4)AND(SAYAR=1) THEN OUTTEXTXY(X+47,Y-33,ST);
   IF (SAYAC=4)AND(SAYAR=2) THEN OUTTEXTXY(X+47,Y-19,ST);
   IF (SAYAC=4)AND(SAYAR=3) THEN OUTTEXTXY(X+47,Y-5,ST);
   IF (SAYAC=4)AND(SAYAR=4) THEN OUTTEXTXY(X+47,Y+9,ST);
 END;
END;
PROCEDURE INTSTR(DEGIS:INTEGER;VAR STR:STRING);
BEGIN
 CASE DEGIS OF
  1:STR:='1';2:STR:='2';3:STR:='3';4:STR:='4';5:STR:='5';
  6:STR:='6';7:STR:='7';8:STR:='8';9:STR:='9';10:STR:='10';
  11:STR:='11';12:STR:='12';13:STR:='13';14:STR:='14';
  15:STR:='15';16:STR:='16';17:STR:='17';18:STR:='18';
  19:STR:='19';20:STR:='20';21:STR:='21';22:STR:='22';
  23:STR:='23';24:STR:='24';25:STR:='25';26:STR:='26';
  27:STR:='27';28:STR:='28';29:STR:='29';30:STR:='30';
```

```
   31:STR:='31';32:STR:='32';33:STR:='33';34:STR:='34';
   35:STR:='35';36:STR:='36';37:STR:='37';38:STR:='38';
   39:STR:='39';40:STR:='40';41:STR:='41';42:STR:='42';
   43:STR:='43';44:STR:='44';45:STR:='45';46:STR:='46';
   47:STR:='47';48:STR:='48';49:STR:='49';50:STR:='50';
   51:STR:='51';52:STR:='52';53:STR:='53';54:STR:='54';
   55:STR:='55';56:STR:='56';57:STR:='57';58:STR:='58';
   59:STR:='59';60:STR:='60';61:STR:='61';62:STR:='62';
   63:STR:='63';64:STR:='64';65:STR:='65';66:STR:='66';
   67:STR:='67';68:STR:='68';69:STR:='69';70:STR:='70';
   71:STR:='71';72:STR:='72';73:STR:='73';74:STR:='74';
   75:STR:='75';76:STR:='76';77:STR:='77';78:STR:='78';
   79:STR:='79';80:STR:='80';81:STR:='81';82:STR:='82';
   83:STR:='83';84:STR:='84';85:STR:='85';86:STR:='86';
   87:STR:='87';88:STR:='88';89:STR:='89';90:STR:='90';
   91:STR:='91';92:STR:='92';93:STR:='93';94:STR:='94';
   95:STR:='95';96:STR:='96';97:STR:='97';98:STR:='98';
   99:STR:='99';100:STR:='100';
  END;
END;
PROCEDURE PARCA(CE,T:INTEGER;VAR CircuitElement:STRING);
BEGIN
 IF T=1 THEN
 BEGIN
  SETVIEWPORT(3,3,155,19,TRUE);CLEARVIEWPORT;
  SETVIEWPORT(0,0,639,479,TRUE);SETFILLSTYLE(1,7);
  SETCOLOR(8);FLOODFILL(5,5,8); SETTEXTSTYLE(2,0,4);
  CASE CE OF 1..23:SETCOLOR(11) ELSE SETCOLOR(5) END;
  MOUSEPART(CE,CircuitElement);OUTTEXTXY(25,5,CircuitElement);
 END;
END;


END.
```