LOJİK DEVRE BENZETİMİ

Gözde ÖRGÜ - Büşra ERKAN

Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

gozde.orgu@gmail.com busraerkan39@gmail.com

Özet

Lojik Devre Benzetimi ile basit devre elemanlarından oluşan mantıksal bir devre, kapı yayılım gecikmeleri dikkate alınarak simule edilir. Kullanıcı programı ilk çalıştırdığında program içindeki satırları yorumlayıp komut çalıştırabilir hale gelir. Kullanıcıdan Y,I,H,L,S,G,G,C komutları alınır. Komut dizisi "komut.txt" dosyası içinde alt alta yer alabilir. Komutlara bağlı işlemler icra edilir. Devre işlemleri kapılara göre yapılır. Bütün komutlar ve komutlara bağlı gerçekleşen değişimler log.txt dosyasına yazdırılır.*

1.Giriş

Lojik Devre Benzetimi basit bir devreyi,kapı yayılım gecikmelerini dikkate alarak simule eden programdır.

Program başlangıcında kullanıcıdan komut girmesi beklenir. Girilen komutlar Y,I,H,L,S,G,G* ya da C olmalıdır. Girilen komut Y ise "devre.txt" dosyasından devre yüklenir ve konsolda gösterilir; komut I ise "değer.txt" dosyasının içindeki değerlerle devre ilklendirilir; komut H ise ilgili uç/uçların değerleri 1 yapılır; komut L ise ilgili uçların değerleri 0 yapılır; komut S ise devre simule edilir; komut G ise ilgili uç/uçların değerlerindeki değişim gösterilir; komut G* ise tüm uçların değerlerindeki değişim gösterilir; komut C ise benzetimden çıkış yapılır.

İlk olarak kapı işlem sırası belirlenir. Kullanılacak kapı tipleri: AND, OR, XOR, NOT, NAND, NOR'dur. Kapıların işlemleri, her kapının kendi fonksiyonunda yapılır.

Devrede yapılan bütün işlemler "log.txt" dosyasına kaydedilir.

2.Temel Bilgiler

Program C dilinde geliştirilmiş olup tümleşik geliştirme ortamı olarak Dev-C++ kullanılmıştır.

3.Tasarım

Lojik Devre Benzetimi'nin programlanma aşamaları aşağıda belirtilen başlıklar altında açıklanmıştır.

3.1 Algoritma

Program çalıştırılmadan önce programın bulunduğu klasörde "devre.txt","değer.txt","başka_dosya.txt","komut.txt" bulunmalıdır. Önceden bulunan bu dosyalar ve "log.txt" açılır. Devredeki uçların ilk değerleri okutulur. Kapı bilgileri alınmadan önce bilgilerin yazmaya başladığı yer bulunur ve kapı bilgileri alınır. Kapıların bilgilerinin okutulmasında giriş sayısı bilinmediğinden kapının giriş bilgileri nanosaniye değerine kadar okutulur. Kullanıcıdan komut girilmesi istenir. Girilen komutlara göre fonksiyonlar çağırılır. Örnek olarak kullanıcı "h" ya da "H" komutu girdiğinde fonksiyon çağırılır ve ilgili uç/uçların değeri 1 yapılır. Daha sonra değişen uç/uçların etki ettiği kapılarla işlem yapmak için kapıların öncelik sıraları belirlenir ve işleme devam edilir. "s" ya da "S" komutu girildiğinde de değişimler simule edilir. Kullanıcı "c" ya da "C" komutu girene kadar program çalışır.

3.2 Kullanılan Fonksiyonlar

*void kapiNAND(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin NAND kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiXOR(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin XOR kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiNOR(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin NOR kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiAND(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin AND kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiOR(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin OR kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiNOT(struct lojik *d,int j,int z,int x,char islemKapi[5][5],int k,int dizi[])*

Devredeki işlemlerin NOT kapısına göre yapılmasını sağlayan fonksiyondur.

*void kapiIslem(struct lojik *d,char kapi1[5][5],char kapi2[5][5],char kapi3[5][5],int j,int z,int x,int k,int dizi[])*

Kapıların işlem sırasını belirleyen fonksiyondur.

void komutY(char komut[],char uc[])

“y” ya da “Y” komutu ile “devre.txt” dosyasından devreyi yükleyen ve konsolda gösteren fonksiyondur.

*void komutI(char komut[],struct lojik *d,char uc[])*

“i“ ya da “I” komutu ile değer.txt” dosyasının içindeki değerlerle devreyi ilklendiren fonksiyondur.

*void komutH(char komut[],struct lojik *d,char uc[],int k,int dizi[])*

“h” ya da “H” komutu ile seçilen ilgili uç/uçları Lojik-1 yapan fonksiyondur.

*void komutL(char komut[],struct lojik *d,char uc[],int k,int dizi[])*

“l” ya da “L” komutu ile seçilen ilgili uç/uçları Lojik-0 yapan fonksiyondur.

*void komutS(struct lojik *d,int dizi[],char komut[])*

“s” ya da “S” komutu ile devreyi simule ederek nanosaniyelere göre uçların değerlerindeki değişimi gösteren fonksiyondur.

*void komutG(struct lojik *d,char uc[],int k,char komut[])*

“g” ya da “G” komutu ile ilgili uç/uçların değerlerindeki değişimi gösteren fonksiyondur.

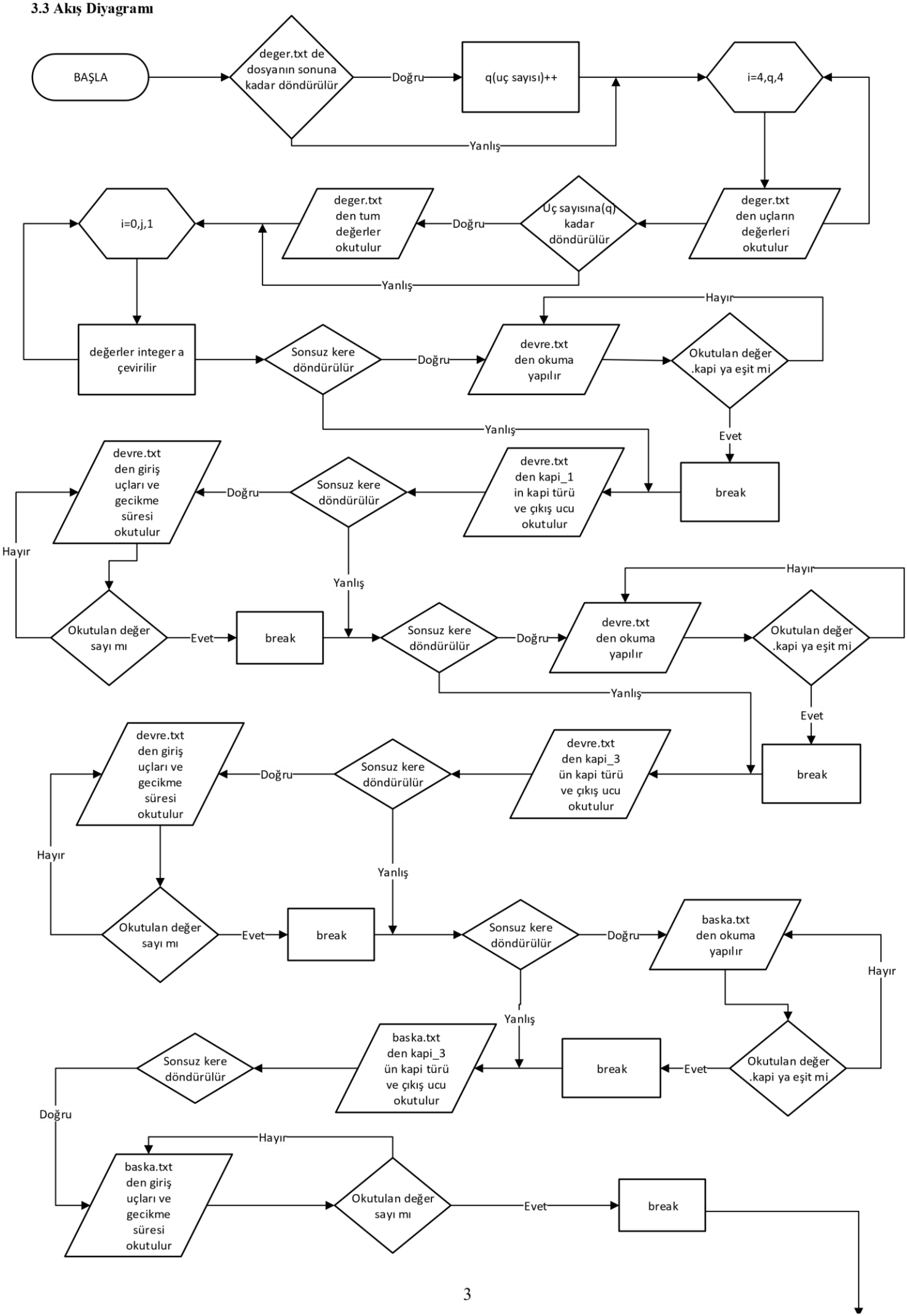
*void komutGyildiz(struct lojik *d,int k,char komut[])*

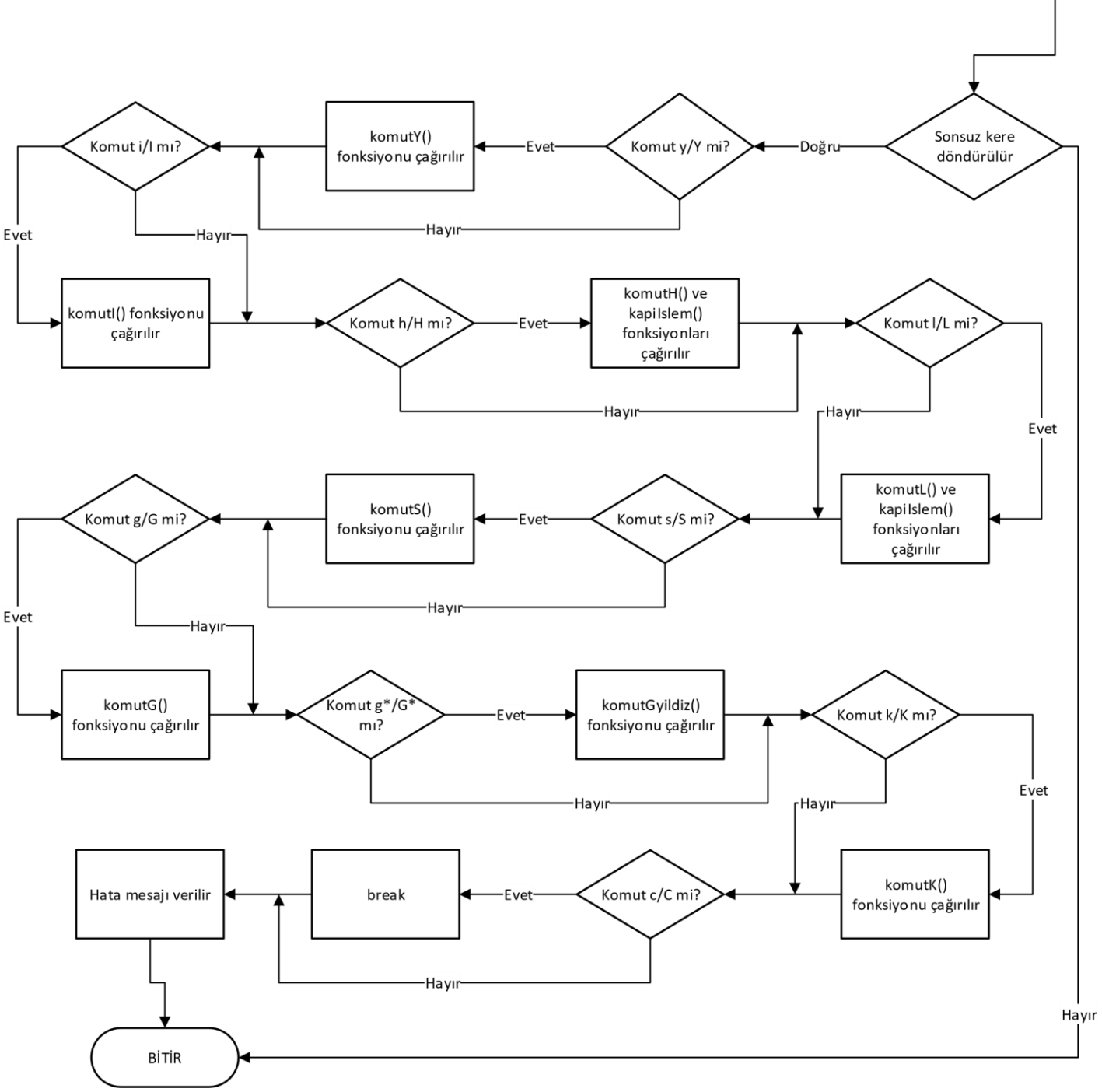
“g*” ya da “G*” komutu ile tüm uçların değerlerindeki değişimi gösteren fonksiyondur.

*void komutK(struct lojik *d,int dizi[],int k,char kapi_1[5][5],char kapi_2[5][5],char kapi_3[5][5],int j,int z,int x)*

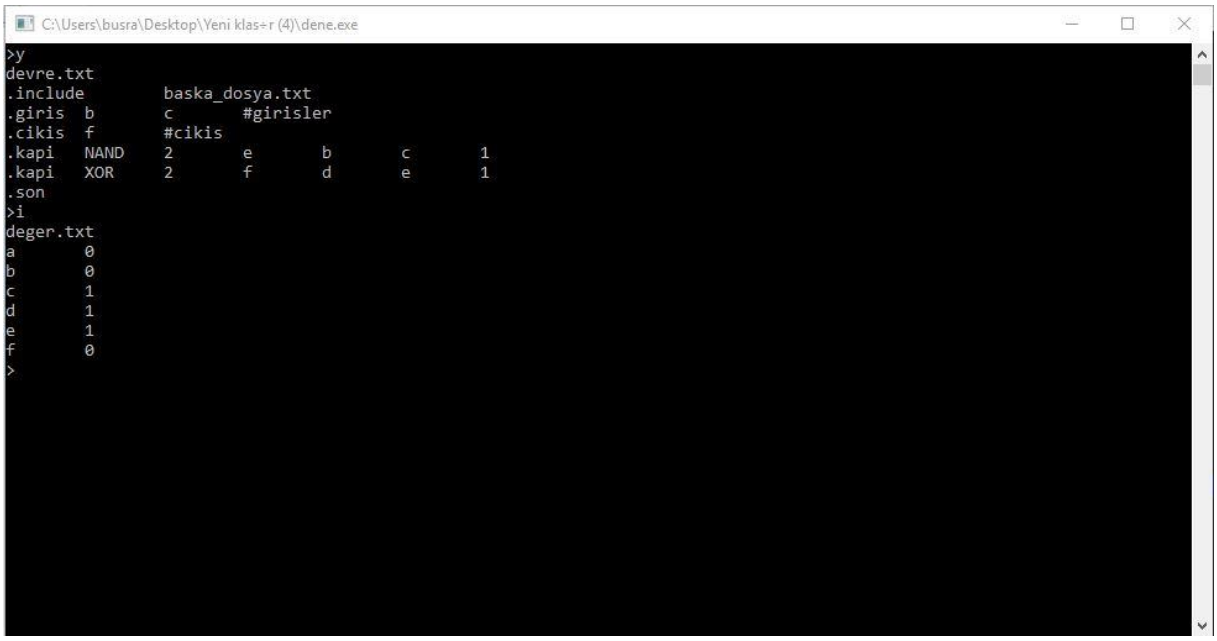
“k” ya da “K“ komutu ile “komut.txt” dosyası içindeki komutları konsoldan uygulayan fonksiyondur.

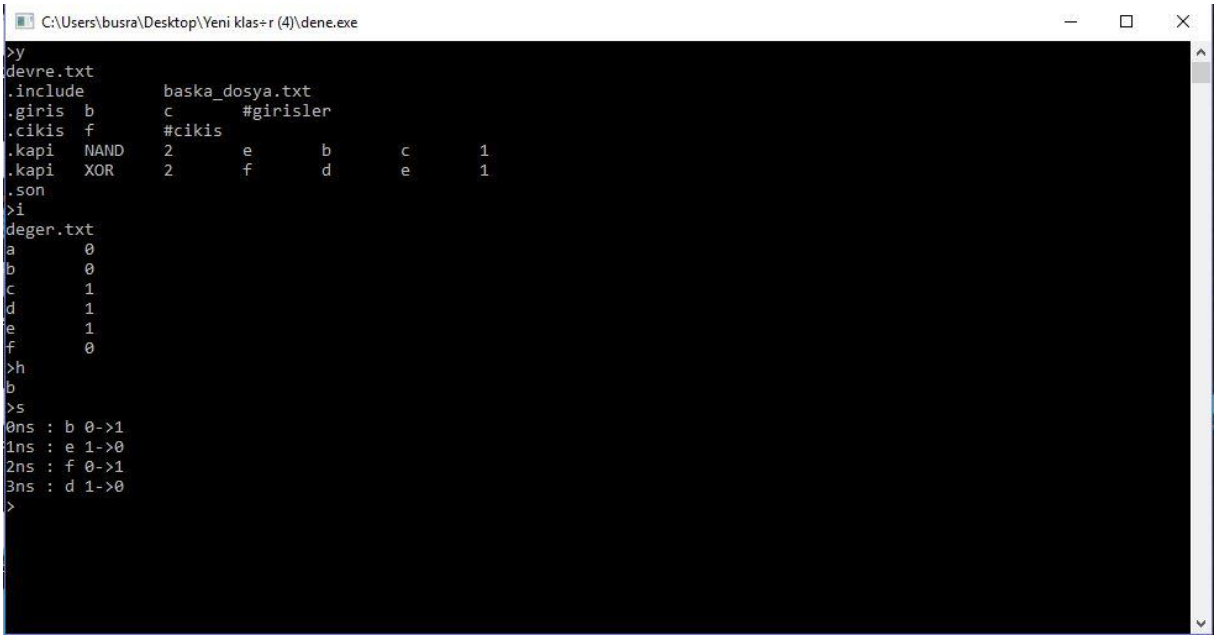
3.3 Akış Diyagramı

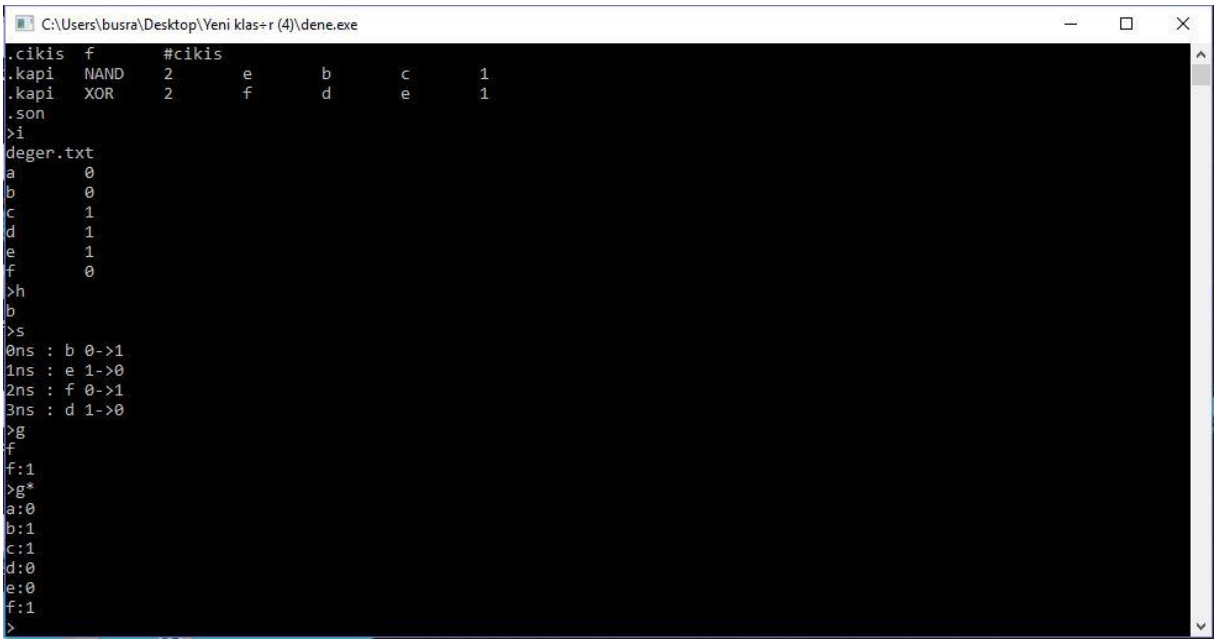


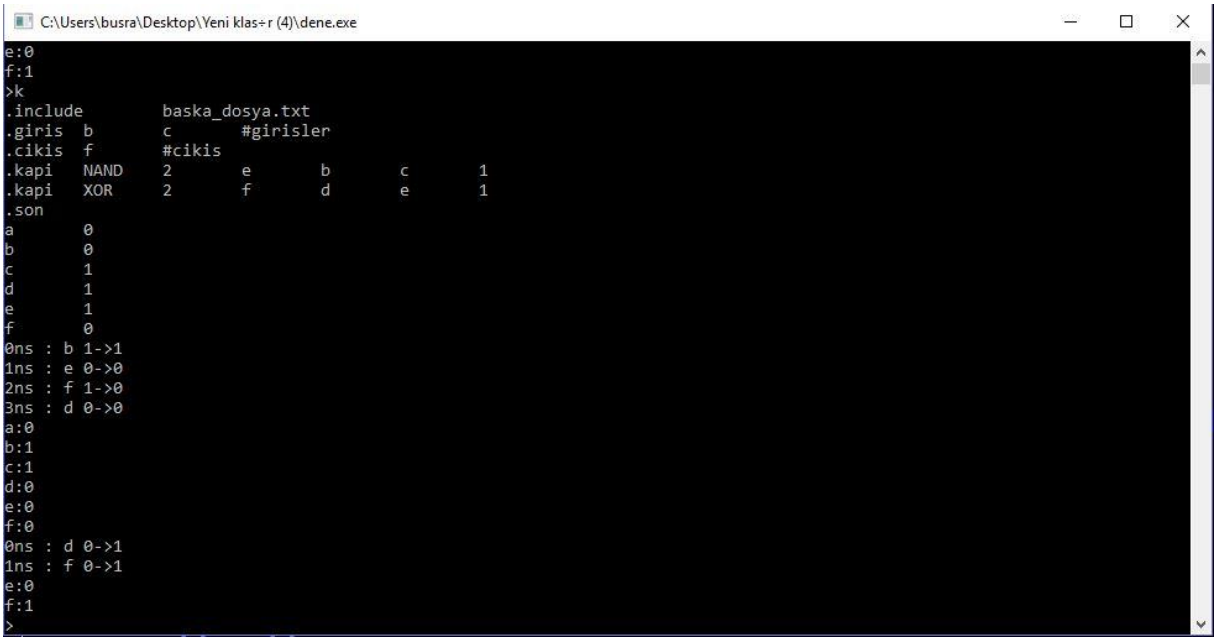

Sonsuz kere döndürülür
Komut y/Y mi?
Doğru
Evet
komutY() fonksiyonu çağırılır
Komut i/I mı?
Hayır
Hayır
Evet
komutI() fonksiyonu çağırılır
Komut h/H mı?
Evet
komutH() ve kapiIslem() fonksiyonları çağırılır
Komut l/L mi?
Hayır
Hayır
Evet
komutL() ve kapiIslem() fonksiyonları çağırılır
Komut s/S mi?
Evet
komutS() fonksiyonu çağırılır
Komut g/G mi?
Hayır
Evet
komutG() fonksiyonu çağırılır
Komut g*/G* mı?
Evet
komutGyildiz() fonksiyonu çağırılır
Komut k/K mı?
Hayır
Hayır
Evet
komutK() fonksiyonu çağırılır
Komut c/C mi?
Evet
break
Hata mesajı verilir
Hayır
Hayır
BİTİR

4.Ekran Çıktıları









6.Kaynakça

Palme Yayınevi - Programlamayı C ile Öğreniyorum