Lojik devre benzetimi

\*Ceren Genç, \*Berkehan Öztürk

\*Bilgisayar Mühendisliği Bölümü, Kocaeli Üniversitesi

Kocaeli, Türkiye

[cerengenc17@gmail.com](mailto:cerengenc17@gmail.com)

[berkehanozturk98@gmail.com](mailto:berkehanozturk98@gmail.com)

**Anahtar Kelimeler – “devre.txt.”, “komut.txt”, “log.txt”**

Özet – Programımız, .txt uzantılı dosyalardan bilgileri okuyarak mantıksal bir elektrik devresini simule etmektedir. "devre.txt” ve “deger.txt” dosyalarından okuma işlemleri gerçekleştirilir.

1.GİRİŞ

Projemiz, kullanıcı tarafından girilen komutlara göre çalışır. İlk komut “Y devre.txt” , ikinci komut ise “I deger.txt” olmak zorundadır. Gereken komutlar girilmediğinde program kullanıcıya hata verip kapanmaktadır. ” “devre.txt” dosyası “#include” ile başlarsa “baska\_dosya.txt” dosyası içinde tanımlı olan kapı, ana devreye eklenir. Projede tanımlı olan komutlar (K, S, H, L, G, C) istenilen şekilde çalışmaktadır. Komut girişinde harf hassasiyeti yoktur. H komutu ile birlikte girilen uç veya uçların mantıksal değeri 1’e yükseltilir ve etkilenen kapıların sonuçları değiştirilir. L komutu ile birlikte girilen uç veya uçların mantıksal değeri 0’a indirgenir ve etkilenen kapıların sonuçları değiştirilir. Bu komutlarda yükseltme ve indirgeme işlemleri, devrede tanımlı olan kapıların gecikme sürelerine göre sırayla yapılmaktadır. Aynı gecikme süresine sahip olan kapılarda değiştirme işlemleri rastgele yapılmaktadır. S komutu ile L veya H komutunda mantıksal değiştirmelerin hangi sırayla yapıldığı kullanıcıya gösterilir. G komutu ile girilen uç veya uçların mantıksal değerleri kullanıcıya konsol ekranı üzerinden gösterilir. G komutu ile “\*” girişi yapıldığında devrede tanımlı olan tüm uçların mantıksal değeri

kullanıcıya gösterilir. K komutu “komut.txt” dosyası içinden komutları okuyarak konsoldan icra eder. C komutunun girişi ile program sonlandırılır.

2.TEMEL BİLGİLER

Projemiz C dilinde yazılmıştır. Geliştirme ortamı olarak “CodeBlocks” kullanılmıştır. Windows 10 ve Windows 7 işlem sistemleri üzerinde sorunsuz bir şekilde çalıştırılmıştır.

3.1 MİMARİ

Dosyalardan okunan uçlar, struct eleman yapısı içinde mantıksal değerleri ve isimleriyle birlikte tutulmaktadır. Projede toplam 6 adet kapı tanımlıdır(OR,NOR,AND,NAND,NOT,XOR). “devre.txt” dosyası içinden okunan kapılar struct kapı yapısı içinde girişleri, çıkışı ve gecikme süresi ile birlikte tutulur. Devre üzerinde yapılan değişikler bu yapılar içinde değiştirilir. Konsol ekranından girilen komutlar ve devrede yapılan değişimler “log.txt” dosyası içinde gün, ay, yıl, zaman şeklinde tutulur.

3.2 KULLANILAN FONKSİYONLAR

1. *struct eleman():* Mantıksal uçların isimleri ve değerleri bu yapı içinde tutulur.
2. *struct kapı():* Mantık kapıları giriş sayıları, çıkışı, girişleri ve gecikme süreleri ile birlikte bu yapı içinde tutulur.
3. *dosyadanOku():* Y komutu girildiği zaman “devre.txt” dosyası içinden gerekli okumalar yapılarak gerekli struct yapıları içinde tutulur.
4. *Iliklendirme():* I komutu girildiği zaman “deger.txt” dosyası içinden gerekli okumaları yaparak mantıksal uçları değerleri ile eşleştirir.
5. *Higher():* H komutu ile birlikte girilen uç veya uçların değeri 1’e yükseltilir. Uç veya uçların etkilediği mantık kapılarının çıkışları gerekli kurallara göre değiştirilir.
6. *Lower():*L komutu ile birlikte girilen uç veya uçların değeri 0’a indirgenir. Uç veya uçların etkilediği mantık kapılarının çıkışları gerekli kurallara göre değiştirilir.
7. *Goster():* G komutu ile birlikte girilen uç veya uçların değerleri konsol ekranından kullanıcıya gösterilir. “G\*” girişi yapıldığında ise tüm uçların mantıksal değerleri gösterilir.
8. *komutK():* K girişi yapıldığında “komut1.txt” dosyası içinden komutlar okunarak konsoldan icra edilir.
9. *Simulation():* S komutu girişi yapıldığında H ve L komutları çalıştığında gerekli değişimlerin hangi sırayla yapıldığı gösterilir.

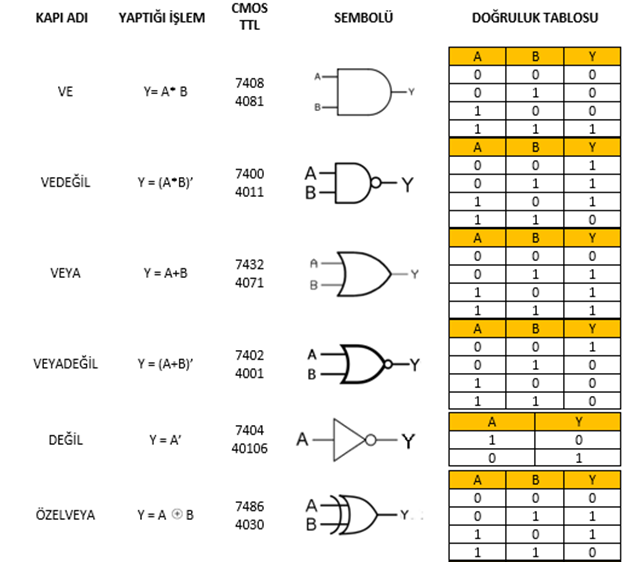
4.SONUÇ

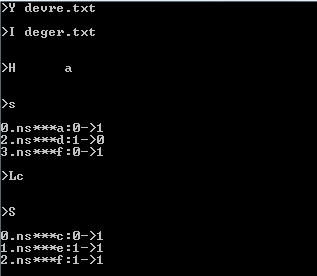
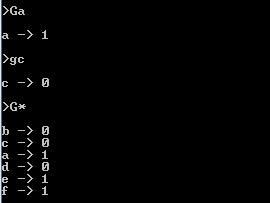
Dosyalar içinde tanımlı olan uçlar, kullanıcıdan istenilen komutlara göre değiştirilir. Uç veya uçların etkilediği kapılar gerekli mantık kurallarına ve gecikme sürelerine göre değiştirilir. Projemiz gerekli olan tüm isterleri yerine getirmektedir.

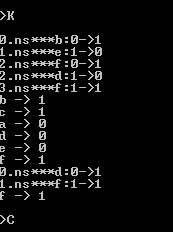
5.KABA KOD

1. Başla
2. Girilecek komutların hangi zamanda girildiğini ve hangi komutun kullanıldığı log1.txt ye yazılır
3. Komutlardan Y ve I girilmezse programdan çıkılır
4. Eğer komut Y ise gerekli parameterler dosyadanOku fonksiyonuna yollanır
5. devre.txt yüklenir
6. Eğer komut I ise iliklendirme fonksiyonu çağırılır
7. deger.txt'deki degerler iliklendirilir
8. Eğer komut H ise istenilen uç veya uçlar Higher fonksiyonuna gereken parametrelerle gönderlir
9. Istenilen ucun logic değeri 1 e yükseltilir
10. Higher fonksiyonu mantıksal operatörlere göre bağlantılı uçları değiştirir
11. Eğer komut L ise istenilen uç veya uçlar fonksiyonuna gönderilir
12. İstenilen ucun logic degeri 0 a indirgenir
13. İstenilen uçların bağlı olduğu kapılar caselerine göre mantıksal işlemlerden geçer
14. Eğer komut g ise istenilen uçların değerini konsol ekranında gösterir
15. Uç olarak \* girilirse bütün devrenin değerleri konsol ekranında gözükür
16. Eğer komut S ise değiştirilen uçlardaki mantıksal operatörlerin hangi nanosaniyede yapıldığını gösterir
17. Eger komut K ise komut.txt deki isterleri okuyup isterlere göre programı çalıştırır
18. Eger komut C ise programdan çıkış yapılır

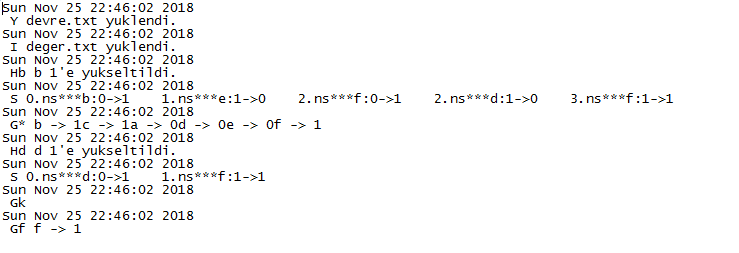
6. EKRAN ÇIKTILARI

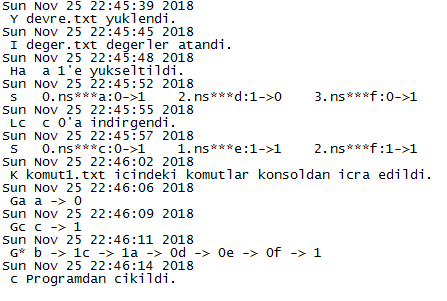
****

**ŞEKİL 1- LOJİK KAPI İŞLEMLERİ**

**** **ŞEKİL 2-KONSOL EKRANI ŞEKİL 3-KONSOL EKRANI**

**ŞEKİL 4- K KOMUTU**

****

**ŞEKİL 5-“log.komut1.txt”**

**ŞEKİL 6-“log.txt”**

7.KAYNAKÇA

<http://kadirselen.blogspot.com/2014/01/tarih-ve-saat-fonksiyonlar.html>