Mantıksal İfadelerin Karnough Haritası Yöntemiyle En Basite İndirgenmesi için Bir Yazılım Geliştirilmesi

Fırat Yücel[1,2], Fatih Arıcı[2], Yusuf Kurt[2]

[1] Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik-Bilgisayar Eğitimi ABD, 32260, Isparta

[2] Serik Endüstri Mes. Lis. ve And. Mes. Lisesi, Bilgisayar Bölümü, 07500, Serik/Antalya

fyucel@ieee.org, fatih_arc@hotmail.com, yusufkurt90@gmail.com

Özet: Sayısal devrelerin tasarımında, devrede kullanılan mantıksal elemanların niceliğinin ve bağlantı sayısının en aza indirilmesi, tasarımın ekonomikliği ve işlevselliği açısından büyük önem taşımaktadır. Bunun sağlanabilmesi için tasarımdaki mantıksal ifadelerin değişik yöntemlerle en basite indirgenmesi gereklidir. Bu çalışmada, mantıksal ifadelerin Karnough haritası yöntemiyle en basite indirgenebilmesi için bir algoritma geliştirilmiş ve Turbo Pascal dilinde kodlanmıştır. Hazırlanan fonksiyonel yapıdaki yazılım, Karnough haritasının analizi için tarama ve gruplandırma algoritmaları içermekte, değişik durumlar için birçok koşul tanımlamaya gerek kalmaksızın zeki kararlar verebilmekte, zaman ve bellek tasarrufu sağlayarak doğru bileşkeleri oluşturmaktadır. Çözümlemede çarpımların toplamı biçiminde indirgeme yapılmıştır. Arama ve gruplandırma, karşılaştırma ve sembolize etme özellikleri görsel öğelerle desteklenmiştir. Tasarlanan yazılım, dört değişkene kadar olan mantıksal ifadeler üzerinde işlem yapabilmektedir.

Anahtar Kelimeler: Karnough Haritası, mantıksal ifade, indirgeme, Turbo Pascal.

A Software Development to Simplify the Logic Expressions By Karnough Map Method

Abstract: In logic circuits design, count of the logic devices used in circuit and connections minimization is very important in terms of realizing an economics and functional design. For providing it, the logic expressions in design must be simplified by different methods. In this study, an algorithm is developed for simplification of the logic expressions by Karnough map method and programmed in Turbo Pascal language. The prepared software, which have a functional structure, contain scanning and grouping algorithms for Karnough map analysis, make intelligent decisions for different states without definition of many conditions for different states, and formed correct groups by provide to save time and memory. When solving, we used sum of product simplification. Finding and grouping, comparing and symbolizing features are supported by visual elements. The designed software can be processed on logic expressions pending four variables.

Keywords: Karnough Map, logic expression, simplification, Turbo Pascal.

1. Giriş

Karnough haritası yöntemi, sayısal devrelerin tasarımında kullanılan mantıksal ifadenin en basit şekle indirgenmesi için mantıksal durumların şema üzerinde analizine dayalı bir yöntemdir. Giriş değişkenlerinin sayısı arttıkça ifadelerin sadeleştirilmesinin zorlaştığı bu yöntem, giriş değişkenleri sayısının az olduğu durumlarda iyi sonuç verir. Harita, karelerden oluşan bir şemadır. Her bir kare bir mintermi gösterir. Tasarımcı bu alanlarda uygun bileşkeler alarak en sade ifadeyi elde edebilir.[1,2]

Karnough haritası yönteminde giriş değişkenlerine göre çıkış durumları tabloya aktarıldıktan son-

ra gruplandırmalar belli kurallara uygun olarak gerçekleştirilmelidir. Aksi halde yanlış veya tam indirgenememiş sonuçlar elde edilebilir. Gruplandırmalar yapılırken çoğu kez doğru bileşkeler tasarımcının gözünden kaçabilir. Bu çalışmada, Karnough haritası yöntemiyle en basite indirgenmiş sonucun hızlı bir şekilde elde edilebilmesi için bir algoritma geliştirilmiş ve bu algoritma *Turbo Pascal* programlama dilinde kodlanmıştır.

Benzer bir çalışma, Wood ve Danielson (2000) [3] tarafından eğitim amaçlı olarak Java platformunda yapılmıştır. Ancak, yapılan çalışmada kullanılan applet'in mantıksal ifadeleri en basite indirgemedeki doğruluk oranı düşüktür. Benzer çalışmalarla ilgili olarak Internet üzerinde yapılan taramalarda, ticarî birkaç yazılım bulunmuştur. [4] Bu durumda, geliştirilen algoritmanın *Boolean* ifadelerin bilgisayar tabanlı analizi konusunda akademik bir katkısının olduğu düşünülebilir.

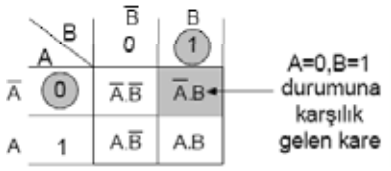
Analiz ve çözümlemenin gerçekleştirilmesi için hazırlanan algoritma, fonksiyonel yapıya sadık kalınarak, oluşabilecek bileşkeleri belirlemede *zeki* kararlar verebilmekte, zaman ve bellek tasarrufu sağlamaya olanak tanımaktadır. Algoritmanın hazırlanmasında, sembolik hesaplamaların daha açık şekilde değerlendirilebilmesi açısından *Turbo Pascal* programlama dili kullanılmıştır.

Karnough haritasının analitik yöntemlerle çözümlenmesindeki öncüller, bilgisayar tabanlı yönteme uyarlanırken matrislerden yararlanılmıştır. Çözümlemede çarpımların toplamı kullanılarak indirgeme yapılmıştır. Arama ve gruplandırma, karşılaştırma ve sembolize etme özellikleri, görsel öğelerle desteklenmiştir.

Tasarlanan yazılım, dört değişkene kadar olan mantıksal durumlar üzerinde işlem yapabilmektedir.

2. Karnough Haritası Yöntemi

Şekil 1'de iki değişkenli Karnough haritası ve karşılık gelen değişkenler gösterilmiştir. Her satır ve sütundaki "1" ve "0"lar değişkenlerin alabileceği durumları göstermektedir. Her bir satır ve sütunun bileşiminden elde edilen ikilik ifade değişkenlerin bulundukları kareye ait durumunu göstermektedir.



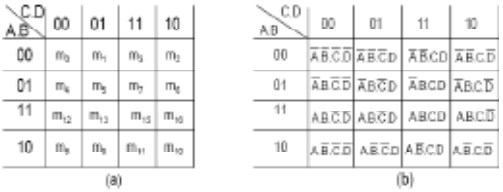
Şekil 1 A=0, B=1 durumuna karşılık gelen karenin gösterimi

Karnough haritası yönteminin indirgemedeki yararını anlamak için birbiriyle bitişik iki kareyi incelemekte yarar vardır. Haritada değişkenin "0" olduğu kare değişkenin değilini, "1" olduğu kare ise değişkenin kendisini tanımlamaktadır. Eğer bitişik iki kareye ait m_1 ve m_3 mintermlerine VEYA işlemi uygulanırsa elde edilen ifade:

$$\begin{aligned} m_1+m_3 &= \overline{A}.B + A.B \\ &= B.(\overline{A}+A) \\ &= B \end{aligned}$$

olacaktır. Bitişik iki kare VEYA işlemine tabi tutulduğunda ifade tek terime indirgenir.

Dört giriş değişkeni için oluşturulan Karnough haritası Şekil 2'de verilmiştir. Dört giriş değişkeni, haritanın on altı kareden ($2^4 = 16$) oluşmasını sağlar. Şekil 2 (a)'da 16 minterm ve yerleşimi gösterilirken, (b)'de ise mintermler Boolean ifadesi şeklinde haritaya yeniden yazılmıştır.



Şekil 2 Dört Değişkenli Karnough Haritası

Doğruluk tablosunda çıkış ifadesi için tercih edilen indirgeme şekline göre "1" veya "0"

olduğu durumlar, Karnough haritasında uygun karelere yazılır.

Mantıksal ifadeleri Karnough haritalarının yardımı ile çarpımların toplamı formunda indirgerken;

1. Bileşke oluştururken içinde "1" olan karelerin sayısı 2^n kadar olmalıdır.
2. Bir kare birden fazla bileşke içinde bulunabilir.
3. Karelerin bileşke oluşturabilmeleri için birbirlerine komşu olmaları gereklidir.
4. Karşılıklı köşe ve kenarlardaki kareler birbirlerine komşu sayılırlar.[1]
5. Bileşke sonuçları VEYA işlemine tabi tutularak indirgeme bitirilir. Bileşke sonuçları belirlenirken;
6. Bileşke içinde durum değiştiren değişkenler varsa (1'den 0'a veya 0'dan 1'e) bu değişkenler dikkate alınmaz.
7. Bileşke içindeki karelerinde durum değiştirmeyen değişkenler varsa indirgemede bu değişkenler dikkate alınır. Durum değiştirmeyen değişkenler *lojik–0* ise değişkenlerin değili, *lojik–1* ise değişkenlerin kendisi yazılır.[1]

3. Karnough Haritası Yöntemiyle En Basite İndirgeme Algoritması

3.1 Veri Yapısının Oluşturulması

Karnough haritasının içeriği, 4x4 boyutlarındaki bir matrise (*Dizi*) kaydedilmiştir. Gruplandırmanın olup olmadığının tespiti için ise her bir hücreyle eşleştirilebilecek şekilde 4x4 boyutlu başka bir matris (*Grup*) kullanılmıştır. Bu matrise, gruplandırma olduğunda "1", olmadığında "0" değeri atanmaktadır.

Haritadaki her bir satır ve sütunun koordinat değişimlerine göre mantıksal değişken değerlerinin değişimini belirlemek üzere aşağıdaki gibi bir sabit tanımlanmıştır[5]:

```
Const
     Dogruluk: Array[0..3,0..1] of
Byte = ((0,0),
        (0,1),
        (1,1),
        (1,0));
```

Dört değişkenli bir Karnough haritası için, sabitte yer alan dizinin ilk boyutundaki değerler *A* ve *C* değişkenlerinin değişimini, ikinci boyutundaki değerler ise *B* ve *D* değişkenlerinin değişimini belirlemektedir. Hücrelerin koordinatları, dizinin iki boyutlu indisi ile ilişkilendirilmiştir. Dolayısıyla, koordinat değerlerinin dizide saklanmasına gerek yoktur. Örneğin ilk hücre için *Dogruluk* dizisinde *A* ve *C* için [0,0] ve *B* ve *D* için [0,1] indisli değerlere bakıldığında tüm değişken değerlerinin 0 olduğu görülür.

3.2 Grupların Tespiti

Hazırlanan yazılım, dört değişkene kadar olan mantıksal ifadeleri analiz etmekte ve indirgemektedir. Amaç, oluşabilecek en büyük bileşkeyi bulmak olduğundan, bileşkeleri tarama için izlenen sıra aşağıda belirtilmiştir:

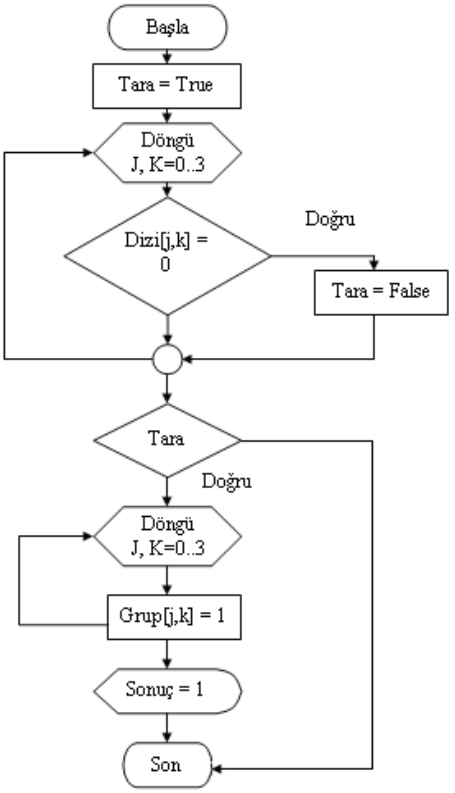
1- 2^4 bileşkesi
2- 2^3 bileşkesi
3- 2^2 bileşkesi
4- 2^1 bileşkesi
5- 2^0 bileşkesi

Karnough haritasında 16 hücreli bileşkenin belirlenmesi için Şekil 3'teki algoritma kullanılmıştır.

Şekil 3'teki algoritmada görüldüğü üzere, 16 hücrenin tamamının "1" olup olmadığı denetlenmekte, gruplandırma gerçekleştirilirken gruplanan hücreler *Grup* adlı diziye işaretlenmektedir. Haritadaki hücrelerin tamamı "1" değerini aldığında indirgenmiş sonuç 1 olmaktadır.

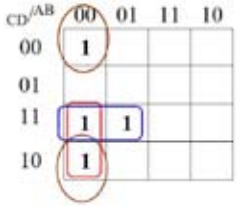
Diğer bileşkelerin (2^3, 2^2, 2^1 ve 2^0) belirlenmesinde, bu algoritmadan farklı olarak, *Grup* matrisindeki değerler kontrol edilmektedir. İlk döngüde *Dizi* içerisindeki bir bileşkeyi oluşturan bütün hücreler taranmakta ve *Grup* matrisindeki ilgili hücrelerin değerlerinden herhangi birisi 0 ise

(gruplandırılmayan hücre varsa) o takdirde gruplandırma işlemi gerçekleştirilmektedir.



Şekil 3 On altı hücreli bileşkeyi belirlemek için kullanılan algoritma

3.3 Aynı Bileşke için Birden Fazla Gruplamanın Önlenmesi



Şekil 4 Aynı Bileşkeyi Birden Fazla Gruplama Hatası

2^2 ve 2^1 hücreden oluşan bileşkelerde, yatay ve dikey olarak gerçekleştirilen taramalar sırasında, bir sonraki hücrenin başka bir bileşkede yer alması halinde aynı bileşke iki kez belirlenmiş olmaktadır. Bu durumda Şekil 4'te verilen Karnough haritasında görüldüğü gibi bir hata oluşabilmektedir.

Şekil 4'te sol alt köşede dikey olarak bulunan ikili grup, birden fazla gruplandırılmıştır. Bu bileşkenin kaldırılması, ifadenin daha basite indirgenmiş biçimde sonuçlanmasını sağlayacaktır. Bu türden bir hatayı engellemek için üç aşamalı bir tarama algoritması kullanılmaktadır:

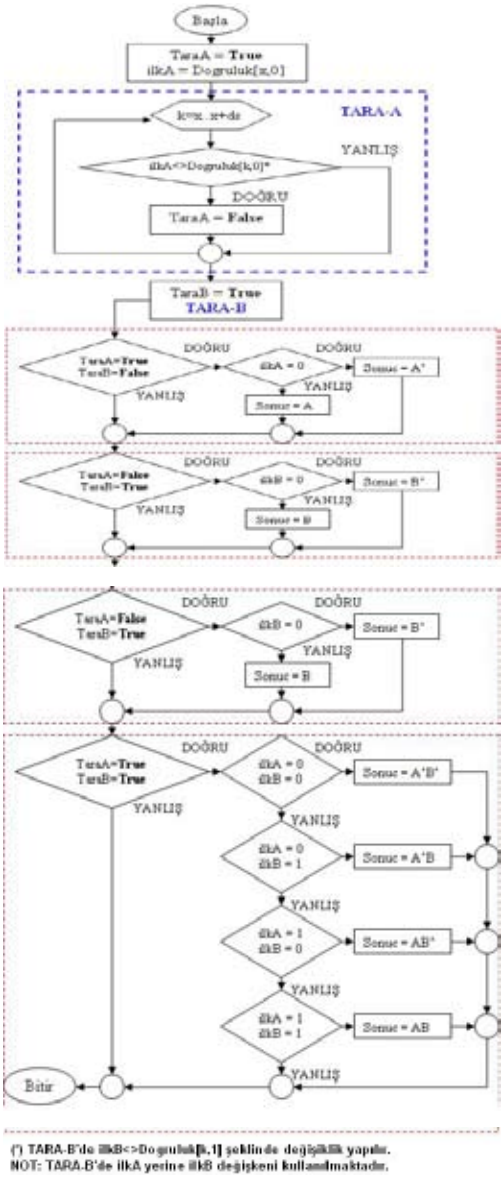
1- İlk önce yapılan taramada, önceden gruplandırılan hücreler üzerinde herhangi bir bileşke taraması gerçekleştirilmemektedir.
2- İkinci taramada bütün hücreler bileşke taramasından geçirilmektedir.
3- Önceki taramada satırlara göre sütunlar taranmışsa, bir sonraki taramada sütunlara göre satırlar taranmıştır.

3.4 Sonuçların Sembolize Edilmesi

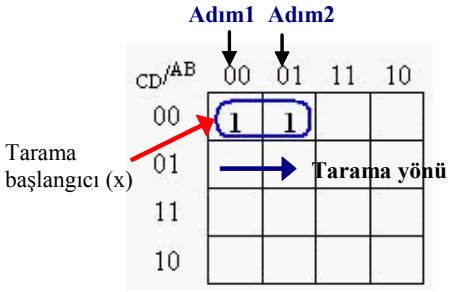
Taramalar sonucunda gruplandırılan bileşkeleri harfler ile sembolize etmek için Karnough haritası üzerinde yatay ve düşeyde tarama yapan iki fonksiyon yapısı kullanılmaktadır. Taramalar, yatayda A ve B değişkenlerini, düşeyde ise C ve D değişkenlerini sembolize etmek için yapılmıştır.

Gruplandırılan bileşkeleri yatayda sembolize etmek için kullanılan algoritma, Şekil 5'te verilmektedir. Bu algoritmanın yer aldığı fonksiyona, tarama başlangıç hücresinin yataydaki konumu (x) ve yatayda taranacak hücre sayısı parametre olarak verilmektedir. "*Dogruluk*" adındaki çok boyutlu dizide, harita değişkenlerinin aldığı değerler ((0,0), (0,1) (1,1), (1,0)) tutulmaktadır. Bu değer kümesinin her bir elemanındaki ilk rakam A değişkeninin aldığı değeri, ikinci rakam ise B değişkeninin aldığı değeri ifade eder. Yatay tarama işleminde, grup boyunca yatay doğrultuda gidildiğinde A ve B değişkenlerinin değer değişimleri kontrol edilmektedir. Örneğin Şekil 6'da x koordinatından başlanarak işaretli grup için tarama gerçekleştirilmektedir. Bu taramada *Adım1*'de A'nın değeri 0'dır. Bu değer algoritmada *ilkA* değişkenine saklanır. Sonra, taramaya devam edildiğinde

Adım2'de A'nın değeri yine 0 olmaktadır. *ilkA* ile karşılaştırma sonucunda değerlerin herhangi birinde değişim görülmemiştir. Eğer herhangi bir değer değişimi yoksa 0 değeri için **A'**, 1 değeri için ise **A** sonucu elde edilir. Şekil 6'da verilen örnekte, A değişkeninin değeri grup boyunca tarama yapıldığında sürekli 0 olarak kaldığından, **A'** sonucu elde edilmektedir.



Şekil 5 Yatayda Sembolize Etme Algoritması



Şekil 6 Sembolize Etme İşlemi

Şekil 5'te verilen algoritmanın *TARA-A* kısmında, A değişkeni için Şekil 6'daki *Adım1* ve *Adım2* taramaları gerçekleştirilmektedir. *TARA-B*'de ise tek fark; B değişkeni için değerlerin yerine konmasını sağlamak amacıyla *Dogruluk* dizisinin ikinci boyutundaki verilerle karşılaştırma yapılmasıdır. Algoritmanın devamında değişim olmayan değerler için uygun harflerle sembolizenin yapılmasını sağlayan koşul ifadeleri yer almaktadır.

3.5 Program Girdileri ve Çıktısı

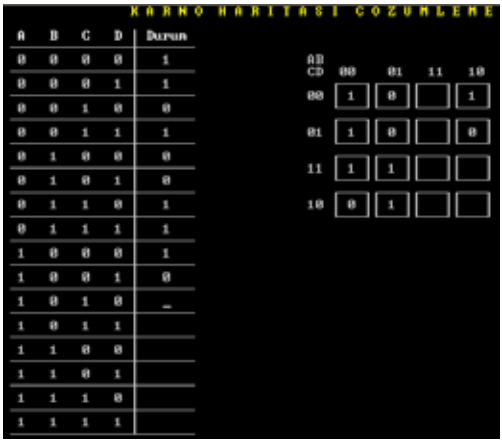
Hazırlanan program çalıştırıldığında, Şekil 7'de görülen mantıksal ifade değişken sayısının sorulduğu bir ekran gelmektedir. Burada girilen değişken sayısı bilgisine göre, bir durum giriş tablosu hazırlanmaktadır.



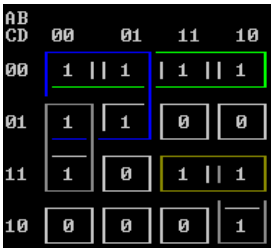
Şekil 7 Giriş Ekranı

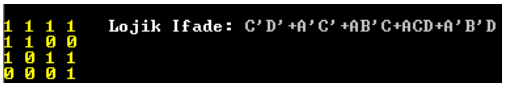
Giriş ekranından sonra, durum girişlerinin yapıldığı Şekil 8'de görülen ekran gelmektedir. Bu ekranda, A, B, C ve D giriş değişkenlerine göre çıkış durumları *lojik-1 (**1**), lojik-0 (**0**)* ya da *fark etmez (**2**)* olarak girilir. Girilen değerler, eş zamanlı olarak Karnough haritası üzerinde işaretlenmektedir.

Çıkış değişkenlerinin girişleri tamamlandıktan sonra, renklendirilmiş şekilde gruplandırmaların yapıldığı ve lojik ifadenin verildiği Şekil 9'da görülen sonuç ekranı gelmektedir.



Şekil 8 Durum Değişkenleri Giriş Ekranı





Şekil 9 Sonuç Ekranı

4. Tartışma ve Sonuçlar

Bu çalışmada, Karnough haritasının bilgisayar ile analizi ve çözümlenebilmesi amacıyla fonksiyonel bir algoritma geliştirilerek, çalışır durumda bir yazılım hazırlanmıştır. Hazırlanan yazılım, Karnough haritasının analizi için tarama ve gruplandırma algoritmaları içermekte, değişik durumlar için birçok koşul tanımlamaya gerek kalmaksızın zeki kararlar verebilmekte ve doğru bileşkeleri oluşturmaktadır. Tasarlanan yazılım, dört değişkene kadar olan *Boolean* ifadelerinin sadeleştirilmesinde eğitim amaçlı olarak da kullanılabilmektedir. Devam eden çalışmalarda, dört değişkenin üzerindeki mantıksal durumlar için en basite indirgeme sağlayan bir algoritmanın geliştirilmesi amaçlanmaktadır.

5. Teşekkür

Yazarlar, bu çalışmanın hazırlanmasında fikirlerinden yararlanılan sayın Yrd.Doç.Dr. Abdülkadir ÇAKIR'a teşekkür eder.

6. Kaynaklar

[1]. Derin, O., *Sayısal Elektronik*, Bölüm 5, Mersin Üniversitesi Web Sitesi, *Mersin Üniversitesi,* 2003.

[2]. Ekiz, H., *Sayısal Elektronik, Mantık Devreleri ve Uygulamaları,*Değişim Yayınları, Adapazarı, 2001.

[3]. Rood, S., Danielson, R., "Java-Based Instructional Materials for Introductory Logic Design Courses", *30th ASEE/IEEE Frontiers in Education Conference*, S2D-10, October 18 - 21, 2000 Kansas City, MO.

[4]. Karnough Map, http://www.puz.com/sw/karnaugh/index.htm, *puz.com.*

[5]. Akgöbek, Ö., *Turbo Pascal ve Programlama Sanatı*, Beta Yayınları, Kırklareli, 1995.