T.C.
MUSTAFA KEMAL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ
ANABİLİM DALI

ELEKTRONİK DEVRELERİN EVRİMSEL ALGORİTMALARLA TASARIMI

MUSTAFA ÇAKIR

YÜKSEK LİSANS TEZİ

ANTAKYA

MAYIS–2007

Mustafa Kemal Üniversitesi

Fen Bilimleri Enstitüsü Müdürlüğüne,

Yrd. Doç. Dr. Ersin ÖZDEMİR danışmanlığında, **Mustafa ÇAKIR** tarafından hazırlanan bu çalışma 14.05.2007 tarihinde aşağıdaki jüri tarafından, **Elektrik-Elektronik Mühendisliği** Ana bilim Dalında **Yüksek Lisans Tezi** olarak kabul edilmiştir.

Başkan: Yrd.Doç.Dr. Ersin ÖZDEMİR İmza…….

Üye : Yrd.Doç.Dr. Emin ÜNAL İmza…….

Üye : Yrd.Doç.Dr. Selçuk MISTIKOĞLU İmza…….

Yukarıdaki imzaların adı geçen öğretim üyelerine ait olduğunu onaylarım.

Kod No:

İmza
14.05.2007

Prof. Dr. Necat AĞCA
Enstitü Müdürü

İÇİNDEKİLER

Sayfa

ÖZET

ELEKTRONİK DEVRELERİN EVRİMSEL ALGORİTMALARLA TASARIMI

Bu tezde, Evrimleşebilen Donanım (**E**volvable **H**ard**W**are-EHW) konusunun **harici** uygulamaları için geliştirilen Genetik Algoritma (GA) ile, ardışık lojik devre tasarımı ve optimizasyonu amaçlanmıştır. Bu amaçla geliştirilen programda tasarlanması istenen devreye ait durum tablosundan yararlanarak, devreye uygulanan girişler, hafıza elemanlarındaki durum değişiklikleri ve devre çıkışına olan lojik etkisi gibi ardışık lojik devrelerin tasarımı için gereken veriler GA'ya aktarılmıştır. Bu veriler doğrultusunda GA, belli bir sayıda çözüm topluluğuna genetik operatörleri sırasıyla uygulayarak belli bir sonlandırma kriterine göre işleyişi tamamlamıştır. Elde edilen çözümler elektronik simülasyon programında denenerek sağlamaları yapılmıştır.

Deneyler için seçilen devrelerin bir kısmında istenen fonksiyon %100'ü sağlanırken, bir kısmında yaklaşık sonuçlar elde edilmiştir. 1 girişli 2 durumlu ve 2 girişli 2 durumlu ardışık lojik devre tasarımlarında durum tablosu %100 başarı ile elde edilirken, 4 girişli 2 durumlu ardışık lojik devre tasarımı için standart GA ile %82.8'lik başarı elde edilmiştir. Bu başarı oranını arttırma amacıyla, *adaptif mutasyon, tufan etkisi* ve *devre yoğunlaşma* gibi soruna özel yaklaşımlar kullanılarak başarı oranında yaklaşık %5'lik bir artış sağlanmıştır. Ayrıca GA tarafından tasarlanan devrelerde, klasik tasarımdan farklı sonuçlar elde edilmiş ve tasarlanan devrelerdeki kapı sayısının azaldığı gözlenmiştir.

2007, 89 Sayfa

Anahtar Kelimeler: Ardışık lojik devre tasarımı, Genetik algoritma, Evrimleşebilen donanımlar

ABSTRACT

DESIGN OF ELECTRONIC CIRCUITS VIA EVOLUTIONARY ALGORITHMS

The aim of this study is to obtain sequential logic circuit design and its optimisation via Genetic Algorithm (GA) which is developed for extrinsic Evolvable HardWare (EHW) application. Firstly via statement table or statement diagrams of circuit that should be designed, necessary data like statement changing that is occured while passing from previous statement through following statement in memory elements, applied inputs to the circuits and effects of these changings to the output of circuit, to design cascade circuit were transferred to GA. Thanks to these data, GA might constitute a certain number of solution classes that have gene number and via appliying genetic operators to this solution classes respectively, according to a certain conclusion criteria and/or generation number, operation of GA was concluded. Obtained solutions and truth of electronic circuit were shown with drawings after trying in simulation program.

Although in this study some of acquired circuits obtained the desired circuits, some of them obtained approximately results. In design of sequential logic circuits which have 1 input 2 states and 2 inputs 2 states, statement table was obtained with rate of 100 %. But in design of sequential logic circuits which have 4 inputs 2 states, traditional GA obtained the statement table with rate of 82.8 %. To increase the rate of success a few approaches were used such as *adaptive mutation*, *cataclysm effect* and *intensify to part of combinational circuit.* Success rate of obtaining the statement table was increased approximately 5 % with these approaches. In some circuits which are desired to design by GA obtained different results when compared with conventional design. In some of designs were reduced the number of logic gates.

2007, 89 Pages

Key Words: Sequential circuit design, Genetic algorithms, Evolvable hardware.

ÖNSÖZ

Yüksek Lisans çalışmamın her aşamasında yardımlarını esirgemeyen, değerli fikir ve katkılarıyla ışık tutan ve yönlendiren danışmanım Yrd. Doç. Dr. Ersin ÖZDEMİR'e, bölüm başkanımız Yrd. Doç. Dr. Emin ÜNAL'a, yüksek lisans derslerimde bilgilerini esirgemeyen Yrd. Doç. Dr. Mustafa ORAL'a, çalışma hayatıma attığım ilk adımdan bu yana desteklerini esirgemeyen değerli yüksekokul müdürüm Yrd. Doç. Dr. Selçuk MISTIKOĞLU'na ve ayrıca manevi katkılarından dolayı eşime çok teşekkür ederim.

SİMGELER VE KISALTMALAR

ASIC	: Özel Uygulama Tümleşik Devresi (Application-Specific Integrated Circuit)
CPLD	: Programlanabilen Hücre Kombinasyonlu Donanım (Complex Programmable Logic Device)
ÇDS	: Çıkış Doğruluk Sayısı
EH	: Evrimsel Hesaplama (Evolutionary Computation-EC)
EE	: Evrimsel Elektronik (Evolutionary Electronics-EE)
EHW	: Evrimleşebilen Donanım (Evolvable HardWare)
EP	: Evrimsel Programlama (Evolutionary Programming-EP)
ER	: Evrimsel Robotik (Evolutionary Robotics-ER)
EWB	: Electronics WorkBench
GA	: Genetik Algoritma (Genetic Algorithm-GA)
GAL	: Genel Lojik Dizisi (Generic Array Logic)
GP	: Genetik Programlama (Genetic Programming-GP)
GSP	: Gezgin Satıcı Problemi
PLD	: Programlanabilen Lojik Donanım (Programmable Logic Device)
PLA	: Programlanabilen Lojik Dizi (Programmable Logic Array)
PROM	: Programlanabilen Sadece Okunur Bellek (Programmable Read Only Memory)
FPGA	: Programlanabilen Dijital Kapı Dizisi (Field Programmable Gate Array)
FPAA	: Programlanabilen Analog Eleman Dizisi (Field Programmable Analog Array)
SDDS	: Sonraki Durum Doğruluk Sayısı
TLKS	: Toplam Lojik Kapı Sayısı
YSA	: Yapay Sinir Ağları (Artificial Neural Networks-ANN)
YZ	: Yapay Zeka (Artificial Intelligence-AI)

ÇİZELGELER DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

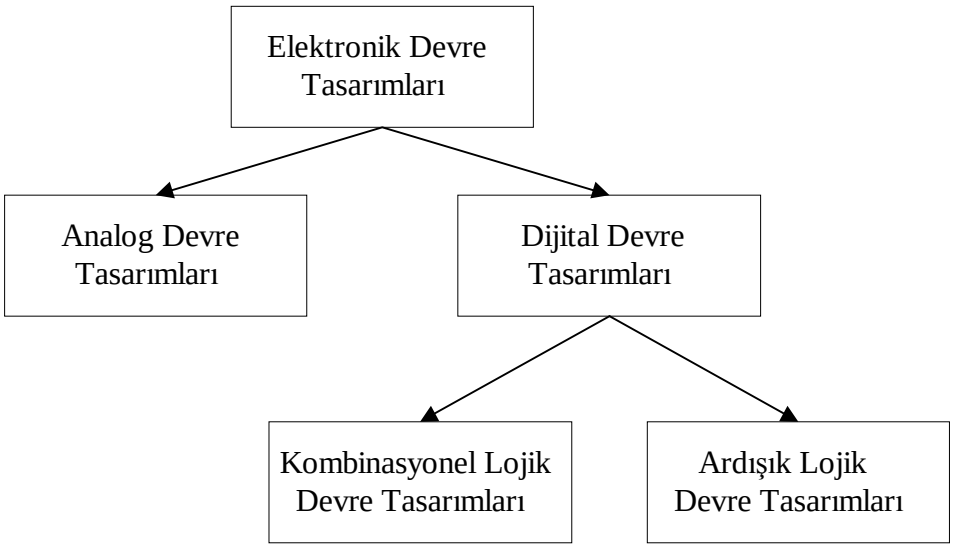
Sayfa

1. GİRİŞ

Bilgisayar teknolojisindeki gelişmeler, bilimsel araştırmaların devre tasarımı alanındaki hızını ve hacmini arttırmıştır. Araştırma alanlarının bir bölümü, insan müdahalesi olmadan görevleri minimum zamanda yerine getirebilecek, kendi kendine hareket eden, yapay zeka (YZ) ürünü akıllı makineler tasarlayıp geliştirmekle ilgilidir.

Tasarım, bir fikrin üretilebilir bir ürüne dönüşmesi işlemidir. Elektronik devrelerin tasarımında bu işlem yazılımlardan ve elektronik elemanlardan faydalanılarak gerçekleştirilir.

Elektronik devre tasarımları analog ve dijital tasarımlar olarak iki sınıfa ayrılabilir. Dijital devreler de kendi içerisinde kombinasyonel ve ardışık olmak üzere iki kısımda incelenebilir. Bu durum Şekil 1.1' de görülmektedir.



Şekil 1.1. Elektronik devre tasarımlarının sınıflandırılması

Kombinasyonel devreler, çıkışında elde edilecek bilginin, tamamen girişlere uygulanan o andaki bilgilere bağlılığı olan devrelerdir. Ardışık lojik devrelerde ise yalnızca girişlere uygulanan bilgiler değil, aynı zamanda hafıza elemanı veya elemanlarındaki önceden elde edilen bilgilere de bağlıdır. Ardışık lojik devrelerde kullanılan hafıza elemanları bir bitlik bilgiyi depolayabilen elemanlardır. Elektronik

terminolojisinde flip-flop olarak da adlandırılan bu elemanlar SR (Set-Reset), D (Data), JK (Jack Kilby[1]) ve T (Toggle) tiplerinde olmaktadırlar.

Devre tasarım tekniği olarak bir çok yaklaşım vardır. Bunlar aşağıda sıralanmıştır (PERKOWSKI, 1986; MILLER ve ark., 2000; ALI ve ark., 2004; AKSOY, 2004).

- Karno haritası (Karnaugh map)
- Quine-McCluskey metodu
- Petrick metodu
- Espresso
- RM (Reed-Muller) lojiği
- Evrimsel Hesaplamalar (EH)

Boolean matematiğinde yapılan sadeleştirmeleri *karno haritası* ile daha kolay ve daha güvenilir yapmak mümkündür. *Karno haritası*, sadeleştirme ve dijital devre tasarımında sıkça kullanılan yöntemlerden biridir. Değişken sayısına göre *karno haritası* düzenlenir. Örneğin 2 giriş (A B), 5 giriş (A B C D E) gibi. *Karno haritası* en fazla 6 girişli *boolean* eşitliklerini sadeleştirmede kullanılır (AMARAL, 2003).

Quine McCluskey ve *Petrick* metotları listeleme yöntemi (tabular method) kullanarak sadeleştirme işlemini 6 değişkenden fazla girişe sahip eşitlikler için gerçekleştirmektedirler. Ancak bu yöntemler 12 den fazla girişe sahip eşitlikler için, kontrol edilmesi zor bir hal almaktadırlar (ALI ve ark., 2004).

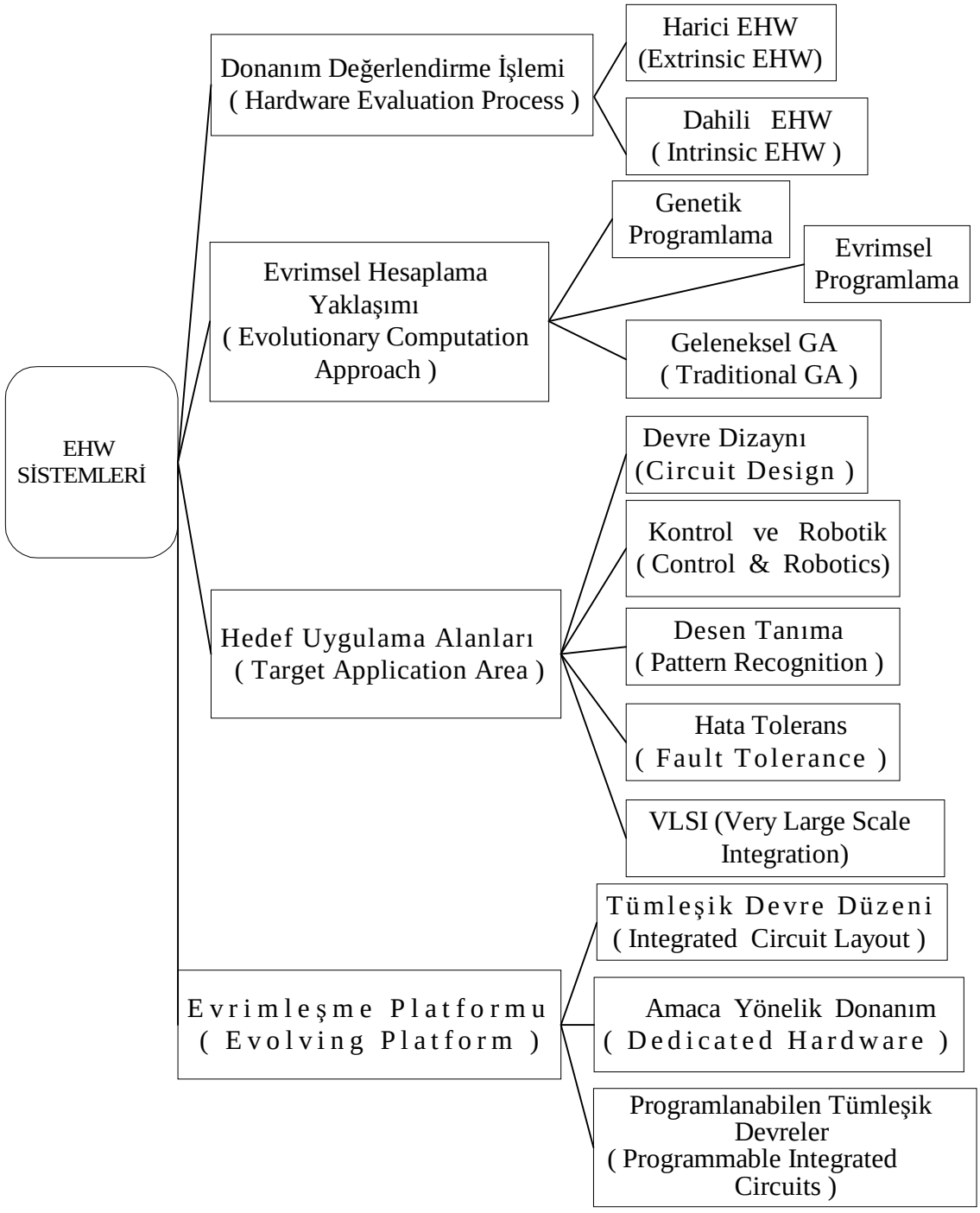
Espresso, doğruluk tablosunda yer alan çıkışlardan "*lojik 1*" durumunu sağlayan durumları kullanarak sade ifadeler arasından en sade ifadeyi elde etmeye çalışan bir algoritmadır. Elde edilen en sade sonucun kalitesini "*lojik 1*" durumunu sağlayan ifadelerin sırası belirler (ZHOU, 2002).

Reed Muller lojiğinde fonksiyonları elde etmede klasik lojik yaklaşımındaki AND ve OR kapılarının yerine EX-OR veya EX-NOR kapıları kullanılmaktadır. Ancak her doğruluk tablosu *Reed Muller* lojiği için uygun olmamaktadır (MILLER ve ark., 2000).

[1] Texas Instruments mühendislerinden tümleşik devrenin mucidi **J**ack **K**ilby'nin adını taşımaktadır.

Elektronik devre tasarımındaki araştırma uzayının karmaşıklığı bizleri Evrimsel metotlar ile devre tasarımı konusunda çalışmalara yöneltmektedir.

Elektronik devre tasarımlarına, evrimsel mühendislik yaklaşımı son zamanlarda yoğun bir şekilde artmaktadır. Programlanabilen tümleşik devrelerin bağlantılarının ve elemanlarının GA kromozomlarına kodlanması, "Evrimleşebilen Donanım (Evolvable Hardware - EHW)" olarak anılan umut verici bir araştırma alanı oluşturmuştur (ZEBULUM ve ark.,1996). Günümüzde EH yaklaşımları, çeşitli alanlarda optimum çözümler sunabilmektedirler. Bu alanlar hali hazırda bir çok araştırmacı tarafından Şekil 1.2'de görüldüğü şekliyle sınıflandırılmaktadır.



Şekil 1.2. EHW sisteminin alt dalları

EHW sisteminin alt dalları (ZEBULUM ve ark., 1996; GORDON ve BENTLEY, 2002) Şekil 1.2' de görüldüğü gibidir ve Bölüm 2 de detaylı bir şekilde incelenecektir.

GA, YZ'nin hızla büyüyen EH'sinin bir alt alanıdır. Bunlar Darwin'in "en iyinin hayatta kalması" prensibini ve evrimin doğal işleyişini taklit eden algoritmalar sınıfındadırlar. EH yaklaşımları arasında GA, dijital devre tasarımlarında kullanılabilmektedir. EHW, elektronik devreleri, doğru veya doğruya yakın bir çıkış elde edebilme yetisine sahip elektronik devrelere evrimleştirmek için GA'dan yararlanır. Her olası elektronik devre, bir bireyi veya çözümü (kromozom) temsil eder. GA, bu çözümlerden oluşturulan topluluk (popülasyon) üzerinde seçim (selection), çaprazlama (crossover) ve mutasyon (mutation) gibi evrimsel işleyişin standart genetik operatörlerini gerçekleştirir. Böylece çevresel değişikliklere karşı dinamik bir adaptasyon sağlanmış olur. Bu durum geleneksel devre tasarımı yaklaşımlarından farklı olarak EHW'nin ayrıcalığını ortaya koyar.

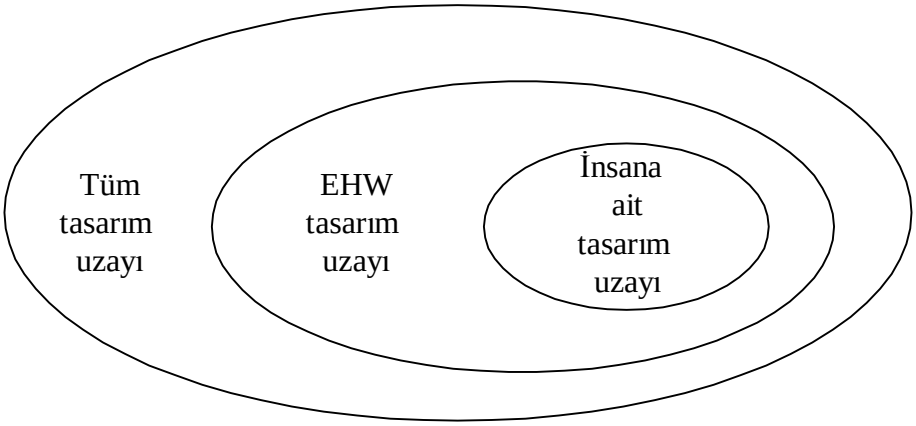
EHW alanında son zamanlarda dikkate değer gelişmeler ortaya konmuş ancak ardışık mantık devreleri üzerine yapılan evrimleştirme çalışmaları henüz tam bir olgunluk göstermemektedir (ALI ve ark., 2004).

Bu tezde GA kullanılarak, kombinasyonel devreler ve D tipi flip-flop'lardan oluşan belirli sayıdaki girişe ve çıkışa sahip ardışık lojik devre tasarımları ve bu tasarımların optimizasyonu amaçlanmıştır.

Çalışmada Borland Delphi 7.0 versiyonu kullanılarak elde edilen programda, belirlenen sayıdaki girişler ve çıkış değerleri ile arzu edilen durum geçişleri için ardışık lojik devreler tasarlanıp optimize edilmiştir.

2. EHW SİSTEMLERİ

EHW, lojik tasarım konusunda alternatif bir yöntemdir. Cebirden bağımsız tekniği ve kendi kendine ürettiği devre ile adapte olabilen ve tekrar ayarlanabilen bir donanımı olduğu için son zamanlarda kullanım alanı artmaktadır.



Şekil 2.1. Tasarım uzayında EHW'nin yeri

Miller ve arkadaşlarına göre EHW tasarım uzayı insan tasarım uzayını kapsamakta olup, tüm tasarım uzayında önemli bir yer tutmaktadır. Bu durum Şekil 2.1' de görülmektedir. EHW tasarım uzayının insan tasarım uzayını kapsaması dolayısıyla geleneksel tasarım metotlarıyla elde edilemeyecek yeni tasarımların elde edilmesi mümkün olabilmektedir (ROGGEN, 2005).

EHW yaklaşımının ortaya çıkması ve gelişimi aşağıdaki kısımda anlatılmaktadır.

2.1. EHW Tarihçesi ve Tanımı

EHW, 1992 yılında İsviçre ile Japonya'da ve 1995 yılında da İngiltere'de ortaya çıkmıştır (ANDERSEN, 1998). EHW, bir YZ araştırmacısı olan Hugo de Garis tarafından 1992'nin yazında ortaya atılmıştır. İlk "Uluslararası EHW Sistemleri Konferansı" (ICES96) 1996 yılında Japonya'da gerçekleştirilmiştir. İkincisi ise 1998 yılında İsviçre'de düzenlenmiştir. EHW alanındaki gelişmeleri konu alan bu tip konferanslar günümüzde de düzenlenmeye devam etmektedir.

EHW'de çevresel değişikliklere adapte olabilmek için kendi devre yapısını ayarlayabilen donanımlardan söz etmek mümkündür (HIGUCHI ve ark., 1999; SAKANASHI ve ark., 1999). Bu donanımlardan ticari açıdan en yaygın dijital donanım FPGA (Field Programmable Gate Array) dır. EHW, kendisini meydana getiren iki ana bileşenden ibarettir. Bunlar, FPGA gibi bir programlanabilen mantık donanımı ve GA gibi bir EH yaklaşımıdır.

EHW'nin nihai hedefi belli bir amacı gerçekleştirecek istenilen bir elektronik devreye takılabilen ve insan müdahalesini en aza indirebilen genel amaçlı bir modül elde etmektir (ZEBULUM ve ark., 1996).

"EHW, çeşitli uygulama alanlarındaki karmaşık devre çözümlerinde kullanılan geleneksel donanım tasarım metotlarına bir alternatiftir. Şu anki inanış, uygun evrimsel donanım dizayn metotlarının, çeşitli alanlarda geleneksel dizayn yaklaşımlarına meydan okuyacak ve güçlü çözümler getirebilme yetisine sahip, yeni EHW sistemlerine yer vereceği yönündedir. Her ne kadar çoğu alan küçük ölçekli problemlere sahipse de EHW, donanım dizaynında daha geniş ölçekli ve gerçek zamanlı uygulamaları gerçekleştirebilmeyi vaat etmektedir" (YAO ve HIGUCHI, 1999).

Bu alanda yapılan çalışmalara bağlı olarak EHW sistemlerini 4 anahtar özellikle karakterize edebiliriz;

1. Donanım Değerlendirme İşlemi
2. Evrimsel Hesaplama Yaklaşımı
3. Hedef Uygulama Alanları
4. Evrimleşme Platformu

Bu maddeler Şekil 1.2' de de yer almakta ve ayrıca sırasıyla aşağıdaki kısımlarda detaylı bir şekilde incelenecektir.

2.2. Donanım Değerlendirme İşlemi

EHW'yi tanımlamanın olası bir yolu da "Genetik Öğrenme (Genetic Learning - GL)" ile tekrar ayarlanabilen donanımların birleşimidir. Donanım değerlendirme işlemi, genetik öğrenme sırasında kromozomların değerlendirildiği ortamın türü ile ilgilenir.

Hali hazırda bu durum, birçok yazar tarafından klasik bir sınıflandırmaya tutulur (POPA, 1999; YAO ve HIGUCHI, 1999; MILLER ve ark., 2000)

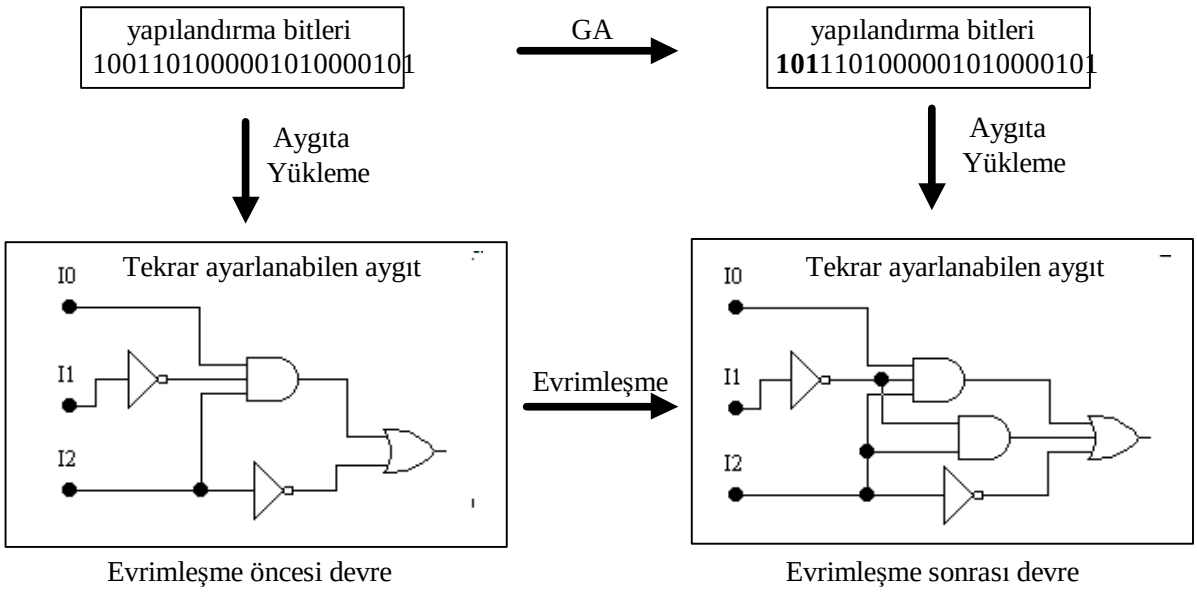
- Dahili EHW (Intrinsic EHW)
- Harici EHW (Extrinsic EHW)

Dahili EHW'deki evrimleşme işleminde herhangi bir simülasyon programı kullanılmazken harici EHW'de "SPICE" gibi bir simülasyon programı kullanılır. Dahili ve harici EHW'ye ait detaylardan sırasıyla aşağıda bahsedilmektedir.

2.2.1. Dahili EHW

Dahili EHW'deki yaklaşım, evrimin tekrar ayarlanabilen donanım içerisinde yer almasıdır. Kromozomlar, tekrar ayarlanabilen donanıma yüklenir ve genellikle donanım mekanizması üzerinden değerlendirilir (ZEBULUM ve ark., 1996; POPA 1999). Bu yaklaşımın iki büyük avantajı vardır. Evrim işleminin hızının yüksek olması ve daha önemlisi devrenin değerlendirilmesi için bir simülasyon modeline ihtiyaç duyulmamasıdır. Aslında bu durumda, bütün çözümler gerçek zamanlı bir uygulamada olduğu gibi davranacaklardır (YAO ve ark., 1999).

Dahili EHW yaklaşımında yapılandırma bitleri kullanılır. Yapılandırma bitleri, tekrar programlanabilen donanım içerisinde oluşacak devreye ait bilgiyi içermektedir. Bu bilgi GA'nın işleyişiyle birlikte değişebilmektedir. Bu değişim farklı bir devrenin tekrar programlanabilen donanıma yüklenmesine sebep olacaktır. Şekil 2.2' de yeniden programlanabilen donanım içerisindeki dijital devreye GA'nın evrimleştirme etkisi yapısal olarak görülmektedir (HIGUCHI ve ark., 1996).



Şekil 2.2. Tekrar programlanabilen donanım içerisindeki devreye GA'nın etkisi

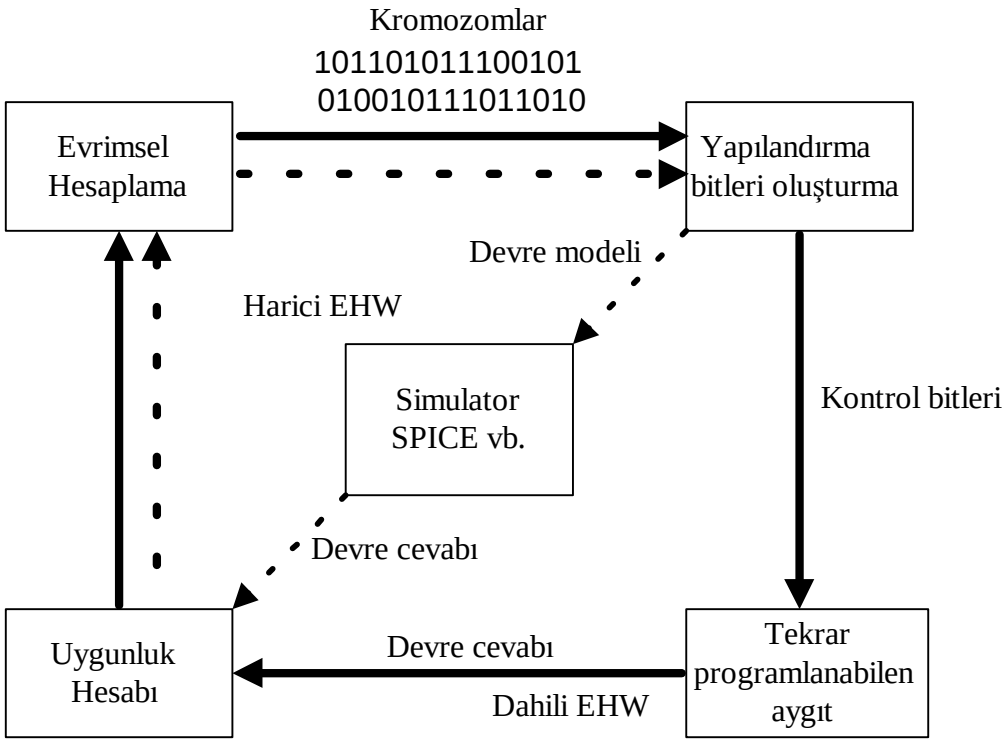
Evrimleşme öncesinde tek bir AND kapısı yer alırken evrimleşme sonrasında ikinci bir AND kapısı devredeki yerini almaktadır. Ayrıca OR kapısına ait giriş sayısı ikiden üçe çıkmaktadır.

İngiltere Sussex Üniversitesinden Adrian Thompson ve Japonya Elektroteknik Laboratuarından (Electrotechnical Laboratory) Tetsuya Higuchi, dahili EHW yaklaşımını kullanarak robotik, desen tanıma ve hata tolerans sistemleri üzerinde çalışma yapanlar arasındadırlar (ZEBULUM ve ark. 1996; ALI ve ark., 2004).

2.2.2. Harici EHW

Çözüm olmaya aday her bir devre kromozomlara kodlanır. Kromozomdaki en basit kodlama ikili kodlama yani 1'ler ve 0'lardan oluşan kodlamadır. Böylesi bir kodlama ile rasgele devreler kromozomlara kodlanır. Harici EHW yaklaşımında, EH'nin oluşturduğu kromozomlar öncelikle simulator için devre modellerine dönüştürülürler. Simulator ile elde edilen devre cevapları ile amaçlanan devre cevabı karşılaştırılarak en iyi devre çözümüne ulaşılmaya çalışılır. Ancak dahili EHW yaklaşımında kromozomların, tekrar ayarlanabilen donanımlara kodlanabilmesi için yapılandırma bitlerine dönüştürülmeleri gerekir. Yapılandırma bitleri oluşan kromozomlar, donanıma yüklenirler ve devre cevaplarına göre sıralanarak hedef devre

cevabına en yakın olan tercih edilir (STOICA ve ark., 2000). Dahili ve harici EHW yaklaşımlarının çalışmasına ait akış, Şekil 2.3' de görülmektedir.



Şekil 2.3. Dahili ve harici EHW'nin karşılaştırılması

Şekil 2.3' de oklarla gösterilen akış takiplerine dikkat edilirse devamlı ok ile gösterilen akış, dahili EHW'yi belirtirken; kesikli ok ile gösterilen akış, harici EHW'nin takip ettiği yolu belirtmektedir.

Harici EHW yaklaşımın dahili EHW yaklaşımından farkı, evrimleşen devrelerin değerlendirilmesi için simülasyon modellerinin kullanılmasıdır. Simulatör olarak "SPICE" gibi programlar kullanılabilmektedir. Evrimleşme işleminin sonunda son çözüm (en iyi çözüm) tekrar ayarlanabilen donanıma yüklenir. Bu yaklaşım daha basittir ve çok sayıdaki bireylerin uygunluk hesabı için elverişli bir yoldur (MILLER ve ark., 2000).

Thompson'un dijital devre tasarımında, harici ve dahili EHW yaklaşımlarının kullanımına dair bir karşılaştırma yer almaktadır. Çalışmasında, hızlı dijital kapılar kullanarak düşük frekanslı osilatör (low frequency osilator) geliştirmiştir (THOMPSON, 1996). Thompson'un yanı sıra harici EHW yaklaşımını Stanford Üniversitesinden bilgisayar uzmanı John R. Koza analog devre tasarımları için, Sony

Bilgisayar Bilimi Laboratuarları (Sony Computer Science Laboratories, Inc.) Yöneticisi Doktor Hiroaki Kitano ise dijital devre tasarımları elde etmek için kullanmışlardır (ZEBULUM ve ark. 1996). Yapılan bu çalışmalar ve detayları ilerleyen kısımlarda anlatılacaktır.

2.3. Evrimsel Hesaplama Yaklaşımı

EHW, devre dizayn etmek için yapay evrimleşme (artificial evolution) kullanır. Ancak, araştırmacıların bunu elde etmek için kullandıkları EH'ler çeşitlilik gösterebilir. Basitçe üç farklı yaklaşım vardır. Bunlar Şekil 1.2' de görülmektedir.

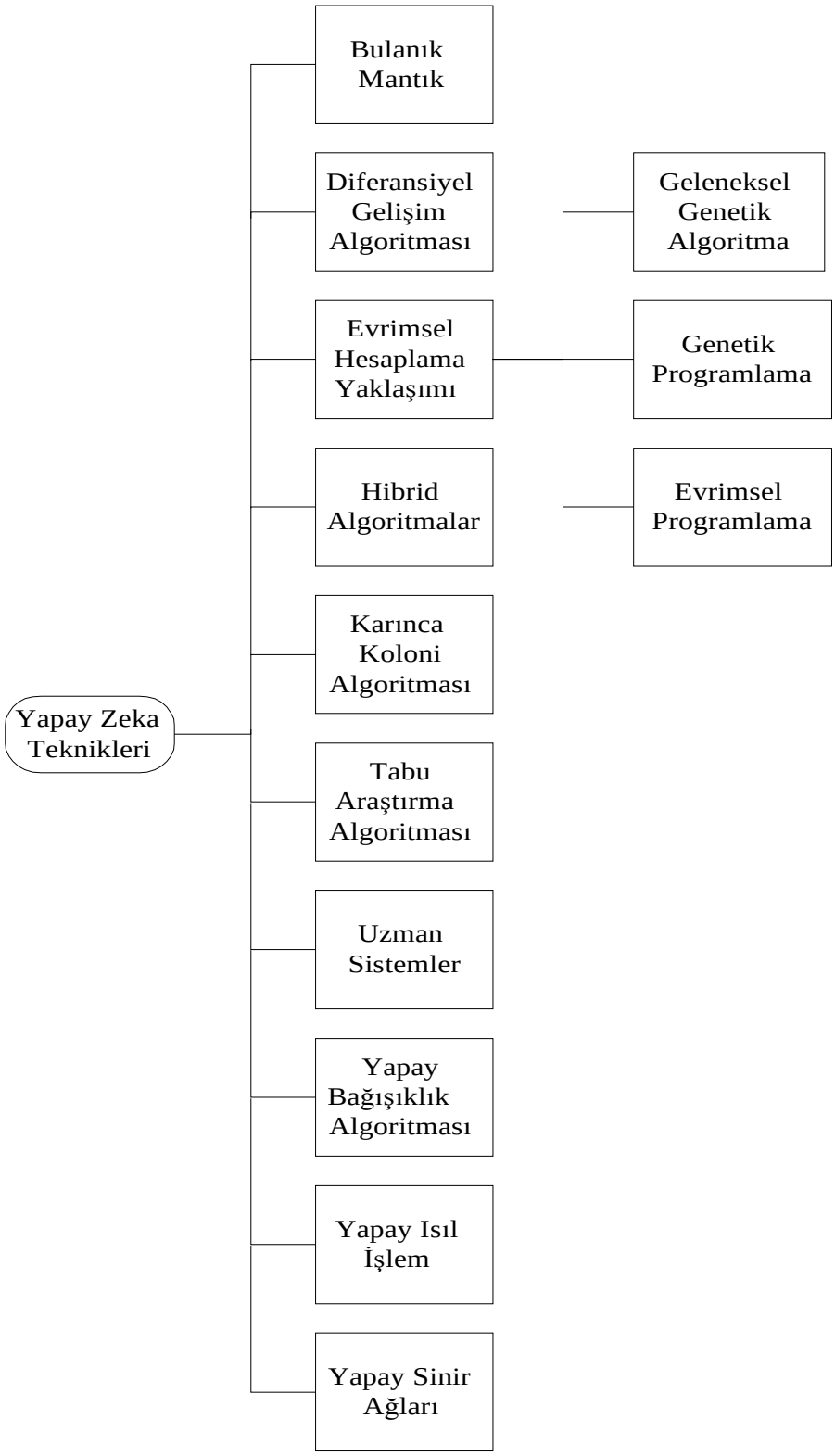
Bunlar sırası ile anlatılacaktır.

2.3.1. Geleneksel Genetik Algoritma

Bilgisayar teknolojisindeki gelişime paralel olarak karşımıza çıkan ve sezgisel olarak çözülebilen, yada çözülmesi matematik teknikleri ile mümkün olmayan problemleri çözmeye yönelik ileri teknikler, YZ teknikleri olarak adlandırılabilirler (TEKTAŞ ve ark., 2007). Bunların başlıcaları Şekil 2.4' de görülmektedir (NABİYEV, 2003; KARABOĞA, 2004; TEKTAŞ ve ark., 2007).

Genel olarak değerlendirildiğinde geleneksel yöntemler, problem ile ilgili bir amaç fonksiyonunun ve gerektiğinde sıralama fonksiyonlarının matematiksel formda tanımlanmasına ihtiyaç duymaktadırlar. Oysaki sezgisel veya YZ teknikleri olarak adlandırılan algoritmalar için böyle bir zorunluluk yoktur (KARABOĞA, 2004).

Şekil 2.4' de görünen YZ tekniklerini değişik kriterlere göre karşılaştırmak mümkündür. Hepsinin birbirine göre avantaj ve dezavantajları vardır. Kullandıkları araştırma stratejilerine göre sınıflandıracak olursak, ısıl işlem ve tabu araştırma algoritmaları komşuluk arama prensibine göre araştırma yapan algoritmalardır. Yani, bir başlangıç çözümü alıp onu iteratif olarak geliştirmeye çalışırlar. EH yaklaşımları, karınca koloni, yapay bağışıklık ve diferansiyel gelişim algoritmaları ise paralel araştırma yapısına sahiptirler. Yani tek bir çözüm alıp onu iteratif olarak geliştirmek yerine bir grup çözüm alarak bu çözümlerden, daha iyi yeni çözüm veya çözümler üretmeye çalışırlar (KARABOĞA, 2004).



Şekil 2.4. GA'nın YZ tekniklerindeki yeri

2.3.1.1. Genetik Algoritmanın tarihsel gelişimi

Michigan Üniversitesinde psikoloji ve bilgisayar bilimi uzmanı olan John Holland bu konuda ilk çalışmaları yapan kişidir. Mekanik öğrenme (machine learning) konusunda çalışan Holland, Darwin'in evrim kuramında etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşünmüştür. Optimum çözümün bulunabilmesi için oluşturduğu kromozom yapılarına seçim, çaprazlama ve mutasyon gibi genetik operatörleri uygulayarak, başarılı yeni bireyler oluşturabildiğini görmüştür. Optimum çözümü bulma için, doğal seleksiyon ve genetik evrimden ilham almıştır. İşlem boyunca, biyolojik sistemde bireyin bulunduğu çevreye uyum sağlayıp daha uygun hale gelmesi örnek alınarak, optimum çözümü bulma ve makine öğrenme problemlerinde, bilgisayar yazılımı modellenmiştir (KOZA, 1995; ALBERT, 1997).

Çalışmalarının sonucunu açıkladığını kitabının 1975'te yayınlanmasından sonra geliştirdiği yöntemin adı GA olarak yerleşti. Ancak 1985 yılında Holland'ın öğrencisi olarak doktorasını veren David E. Goldberg adlı inşaat mühendisi 1989'da konusunda bir klasik sayılan kitabını yayınlayana dek GA'nın pek pratik yararı olmayan bir araştırma konusu olduğu düşünülüyordu. İlk olarak Hollanda'da makine öğrenme sistemlerine yardımcı olarak kullanılmış, daha sonra De Jong Goldberg ve diğerleri tarafından analiz edilmiştir. Goldberg, GA'yı çok sayıda kollara ayrılmış gaz borularında, gaz akışını düzenlemek ve kontrol etmek için kullanmıştır. Ayrıca kendisinin kullandığı makine öğrenmesi, nesne tanıma, görüntü işleme ve işlemsel arama gibi alanlarda kullanılmıştır (WHITLEY, 1994).

Goldberg'in gaz boru hatlarının denetimi üzerine yaptığı doktora tezi ona sadece 1985 National Science Foundation Genç Araştırmacı ödülünü kazandırmakla kalmayıp, GA'nın pratik kullanımının da olabileceğini kanıtlamıştır. Ayrıca kitabında GA'ya dayalı tam 83 uygulamaya yer vererek GA'nın dünyada birçok araştırmada kullanılmakta olduğunu göstermiştir (ALBERT, 1997).

GA, günümüzde de birçok alana uygulanmış ve etkili sonuçlar elde etmeyi başarmıştır. Bu uygulama alanlarına ilerleyen kısımlarda değinilecektir.

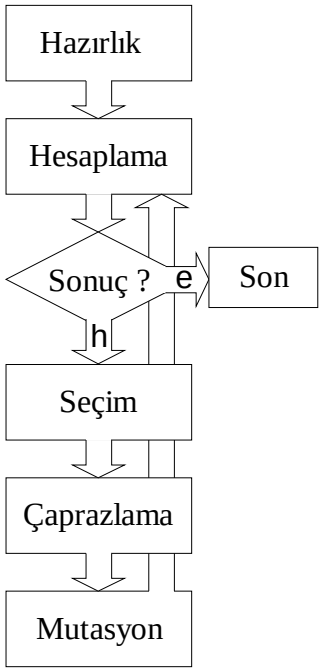
2.3.1.2. Genetik Algoritmanın Tanımı ve Genel Bilgiler

Günümüzün karmaşık ve zor problemleri, hızlı ve kolay çözüm veren yeni çözüm yöntemleri arayışına neden olmuştur. Darwin'in evrim mekanizmasından yola

çıkılarak geliştirilen EH'ler çok boyutlu optimizasyon problemlerini çözmede kullanılan hesaplama stratejilerindendir. Evrimsel yaklaşımlardan olan GA da, bu arayışlar içinde önemli bir yer tutmaya başlamıştır.

Darwin'in, güçlünün hayatta kalması fikrine dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi" nin ne olduğunu belirleyen bir uygunluk fonksiyonu ve yeni çözümler üretmek için çaprazlama, mutasyon gibi operatörler kullanır (ÇAKIR ve OZDEMIR, 2006; KERT, 2006).

GA'nın basit akış diyagramı Şekil 2.5' de gösterilmiştir. GA, hazırlık ile başlayan hesaplama işlemi sonrasında belirlenen hedefe ulaşıp ulaşmadığını test eder. Belirlenen hedefe ulaşılmamış ise seçim algoritmasını, tek noktalı çaprazlama ve mutasyon operatörlerini çalıştırır. Sonuca ulaşıncaya kadar bu işlemler devam eder.



Şekil 2.5. GA'nın basit akış şeması

Her çözüm bir kromozom olarak isimlendirilir. Kromozomlar herhangi bir sayı yada harf sistemi kullanılarak soruna özgü oluşturulabilir. GA, kromozomlardan oluşan çözüm grupları üzerinde istenen çözüme ulaşıncaya kadar çaprazlama ve mutasyon operatörlerini uygular. Şekil 2.5' de görülebileceği gibi, algoritmanın başlangıcında, çoğunlukla rasgele olarak, bir sayı üretme jeneratörü tarafından bir grup çözüm üretilir.

Başlangıç popülasyonunun üretilmesinin ardından, her çözümün uygunluğu belirlenen bir uygunluk fonksiyonu kullanılarak değerlendirilir. Seçim, çaprazlama ve mutasyon gibi genetik operatörler, elde edilen iyi çözümlerden yeni çözümler üretmek için kullanılırlar. Bu iyileştirme işlemi, daha önceden belirlenen bir jenerasyon sayısına veya yeterli bir sonuca ulaşıncaya kadar devam ettirilir.

GA, biyolojik bir sistemin, çevresine adaptasyonunda kullandığı metodun örneklendirilmesidir. Bilgisayarda, bu tür çok parametreli optimumu bulma problemlerine ve makine öğrenme problemlerine çözüm modeli olarak alınabilir (GORDON ve BENTLEY, 2002).

Doğal adaptasyondan esinlenen GA'nın basit olarak iskeleti:

a) Bireyin bulunduğu ortamda hayatta kalmak için, kendi kendisini değiştirerek ortama uygun hale gelmesi,
b) Bu adaptasyon boyunca, yeni üretilecek nesillere, bu özellikler ile birlikte mümkün olabilecek daha çok değişim aktarılarak, bireylerin daha çok uyumlu hale getirilmesi olarak özetlenebilir.

GA'yı diğer algoritmalardan ayıran en önemli özelliklerden biri seçimdir. GA'da çözümün uygunluğu, onun seçilme şansını arttırır ancak bunu garanti etmez. Seçimde seçilme olasılıklarını çözümlerin uygunluğu belirler.

GA'yı diğer metotlardan ayıran noktalar şu şekilde sıralanabilir:

- GA, sadece bir arama noktası değil, bir grup arama noktası (adaylar) üzerinde çalışır. Yani arama uzayında, yerel değil genel arama yaparak sonuca ulaşmaya çalışır. Bir tek yerden değil bir grup çözüm içinden arama yapar.
- GA, arama uzayında bireylerin uygunluk değerini bulmak için sadece "amaç - uygunluk fonksiyonu" (objective-fitness function) ister. Böylelikle sonuca ulaşmak için türev ve diferansiyel işlemler gibi başka bilgi ve kabul kullanmaya gerek duymaz.
- Bireyleri seçme ve birleştirme aşamalarında önceden belirlenmiş kurallar yerine "olasılık kuralları" kullanır.
- Diğer metotlarda olduğu gibi doğrudan parametreler üzerinde çalışmaz. GA, optimize edilecek parametreleri kodlar ve parametreler üzerinde değil, bu kodlar üzerinde işlem yapar. Parametrelerin kodlarıyla uğraşır. Bu

kodlamanın amacı, optimizasyon problemini kombinasyonel bir probleme çevirmektir.

GA, yeni bir nesil oluşturabilmek için 3 aşamadan geçer:

1. Eski nesildeki her bir bireyin uygunluk değerini hesaplama.
2. Bireyleri, uygunluk değerini göz önüne alarak (uygunluk fonksiyonu) kullanılarak seçme.
3. Seçilen bireyleri, çaprazlama, mutasyon gibi genetik operatörler kullanarak çözüm zenginliğine kavuşmuş farklı bireyler elde etme.

GA'ların günümüzdeki uygulamaları üç ana gruba ayrılabilir.

"Bunlardan ilki deneysel uygulamalardır; mevcut diğer optimizasyon algoritmalarına karşı GA'ların üstünlüğünü kanıtlamak amacı ile belirli problemlerin çözümlerini bulmak için GA'ların kullanıldığı çalışmalardır. Bu gruba örnek olarak; Gezgin Satıcı Problemi (Travelling Salesman Problem), Sırt Çantası Problemi (Knapsack Problem), İki-kollu ve K kollu yol kesme (Bandit) problemleri, Grafik Bölme Problemi (Graph Partitioning Problem) gibi çalışmalar gösterilebilir.

İkinci grup pratik uygulamalardır; GA'ların endüstri ve diğer gerçek problemlerin çözümlerinde kullanıldığı uygulamalardır. Bu grup için Nümerik Optimizsyon Problemleri, Çizelgeleme Problemleri (Scheduling Problems), Yerleşim Problemleri (Layout Problems) ve Görüntü İşleme (Image Processing) uygulamaları örnek olarak verilebilir.

Üçüncü ve son grup, sınıflandırıcı sistemler (Classfier Systems) uygulamalarını ihtiva etmektedir. Bunlar, uzman sistemin bilgi tabanını oluşturan kuralları elde etmek için GA'ların kullanıldığı uygulamalardır" (KARABOĞA, 2004).

Geleneksel GA'ların esas avantajı, her bir bilgi ünitesinde maksimum sayıda yapı bloklarının elde edilmesidir. Bu yapı blokları elde edilirken farklı sayı sistemleri kullanılabilmektedir. Ancak ikili sayı sistemi kullanılarak elde edilen kromozom yapısı bazı uygulamalarda geniş yer tutar.

Kromozom yapısının oluşturulması sırasında kullanılan kodlama yöntemlerine aşağıdaki kısımda değinilecektir.

2.3.1.3. Kodlama yöntemleri

Kodlama planı, GA'nın önemli bir kısmını teşkil eder ve hazırlık aşamasında gerçekleştirilir. Bir problemi GA ile çözmeye başladığımızda, karşılaşacağımız problemlerden bir tanesi kromozomların kodlanmasıdır. Bilginin ifade edilmesi kodlama ile gerçekleştirilir. Kromozom genellikle, problemdeki değişkenlerin belli bir düzende sıralanmasıdır. Kromozomu oluşturmak için sıralanmış her bir değişkene "gen" adı verilir. Buna göre bir gen kendi başına anlamlı genetik bilgiyi taşıyan en küçük genetik yapıdır. Mesela; 101 bit dizisi bir noktanın x-koordinatının ikilik düzende kodlandığı gen olabilir. Aynı şekilde bir kromozom ise bir ya da daha fazla genin bir araya gelmesiyle oluşan ve problemin çözümü için gerekli tüm bilgiyi üzerinde taşıyan genetik yapı olarak tanımlanabilir. Örnek vermek gerekirse; 100-011-101-111 şeklindeki bir yapıya sahip kromozom x1, y1, x2, y2 koordinatlarından oluşan iki noktanın konumu hakkında bize bilgi verecektir.

Kromozomların kodlanması probleme göre değişmektedir. Kullanılan kodlama yöntemleri aşağıda sunulmaktadır.

2.3.1.3.1. İkili kodlama

Bu yöntem ilk GA uygulamalarında kullanıldığı için hala en çok kullanılan yöntemlerdendir (NABİYEV, 2003). Burada her kromozom 0 ve 1'lerden oluşan bit dizisidir ve ikili diziyle ifade edilir. Bu dizideki her bit, çözümün bir özelliğini taşır. Dizinin tümü ise bir sayıya denk gelir. Çizelge 2.1, ikili kodlama ile oluşturulmuş iki farklı kromozomu göstermektedir.

Çizelge 2.1. İkili kodlama için kromozom örnekleri

Kromozom 1	0110011010
Kromozom 2	1100001110

Bu kodlama ile bazı problemlerde çaprazlama ve mutasyon sonrasında düzeltmeler yapılması gerekebilir.

2.3.1.3.2. Permutasyon kodlama

Bu kodlama Gezgin Satıcı Problemi (GSP) ve İş Sıralama (Task Ordering) gibi düzenleme problemlerinde kullanılır. Burada her kromozom, bir sayılar dizisidir. Çizelge 2.2, 1 ile 9 arası sayıların bir kez kullanıldığı permutasyon kodlama ile elde edilmiş iki farklı kromozom örneğini göstermektedir.

Çizelge 2.2. Permutasyon kodlamalı kromozom örnekleri

Kromozom 1	4 7 9 1 2 5 8 6 3
Kromozom 2	6 5 8 7 4 3 1 2 9

Belli bir şehirden başlayarak toplam 9 farklı şehiri en kısa yoldan geçme (GSP) gibi bir sorunun GA ile çözülmesi gerektiğinde, her bir şehre bir numara verilip kodlama rasgele yapılırsa Çizelge 2.2' de yeralan permutasyon kodlamalı kromozom yapısı uygun olacaktır.

2.3.1.3.3. Değer kodlama

Reel sayılar gibi tam olmayan değerlerin kullanıldığı problemlerde direkt değer kodlanması kullanılabilir. Örneğin reel sayı, karakter veya bazı kompleks nesneler olabilir. Bu tip problemler için ikili kodlama çok zordur. Çizelge 2.3' de değer kodlamalı üç farklı kromozom örneği yer almaktadır.

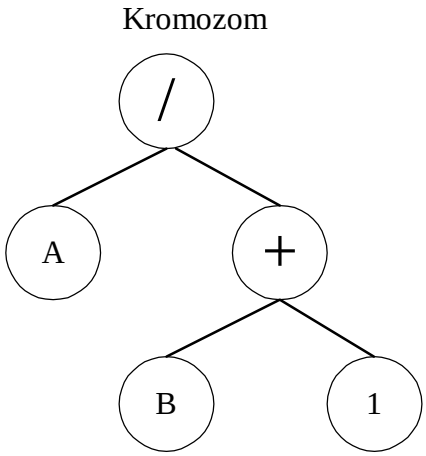
Çizelge 2.3. Değer kodlamalı kromozom örnekleri

Kromozom 1	2.2128 7.3643 0.5457 1.0693 8.4545
Kromozom 2	CBTJKIFJDHDIERJFDLDFLFEGT
Kromozom 3	(ileri), (geri), (sağ), (sol), (geri)

Bu kodlama yöntemi bazı özel problemler için (mesela bir yapay sinir ağının ağırlık katsayısının bulunması) kolaylık sağlar. Diğer taraftan bu tip kodlama için probleme özel yeni bazı mutasyon ve çaprazlamalar geliştirmek gerekebilir (OBITKO, 1998).

2.3.1.3.4. Ağaç kodlama

Ağaç kodlama çoğunlukla Genetik Programlamada (GP), program veya ifadeler için sıklıkla kullanılır. Bu kodlamada her kromozom kendisini oluşturan nesnelerin, bir ağacıdır. Bu nesneler matematiksel fonksiyonlar olabilmektedir. Şekil 2.6' da matematiksel fonksiyonları içeren ağaç kodlamalı bir kromozom örneği yer almaktadır.



Şekil 2.6. (/ A (+ B 1)) kodlamalı ağaç

LISP gibi bir YZ programlama dili bu kodlama metodunu çok kullanır (NABİYEV, 2003).

Bu kodlama metoduna ilerleyen kısımlarda detaylı bir şekilde değinilecektir.

2.3.1.4. Uygunluk Hesabı

Başlangıç topluluğu hazırlık aşamasında bir kez oluşturulduktan sonra evrim başlar. GA, bireylerin uygunluk veya iyiliklerine göre ayrılıp fark edilmesine gerek duyar. Uygunluk, topluluktaki çözümü ifade eden bir kısım bireyin hedef çözüme olan yakınlık derecesidir. Yüksek ihtimalle uygun olan bireyler seçim, çaprazlama ve mutasyon operatörleriyle seçilirler.

Bazı problemler için bireyin uygunluğu, bireyden elde edilen sonuç ile hedeflenen sonuç arasındaki hatadan bulunabilir. Daha iyi bireylerde bu hata sıfıra yakın olur.

Değerlendirme fonksiyonu, her bir kromozomun durumunu değerlendirmeye yarayan ana kaynaktır. Bu, GA ve sistem arasında önemli bir bağlantıdır. Fonksiyon,

kromozomun performansına bağlı olarak bir değerlendirme değeri üretir. Bu, diğer kromozomlar için de yapıldıktan sonra yapıldıktan sonra bu değerler kullanılarak, uygunluk değeri uygunluk fonksiyonuyla hesaplanır.

Uygunluk değerleri hesaplanan çözümler arasında en iyi uygunluğa sahip çözüm veya çözümler yeni nesle aktarılmak üzere saklanırsa bu yönteme “elitzm” adı verilir. Elitizm, en iyi bulunan çözüm veya çözümlerin kaybolmasını önler.

2.3.1.5. Genetik Operatörler

Genetik operatörler, varolan çözüm topluluğu üzerine uygulanan işlemlerdir. Bu işlemlerin amacı, daha iyi özelliğe sahip yeni nesiller üretmek ve arama algoritmasının alanını genişletmektir.

Temelde üç tip genetik operatör vardır. Bunlar; Seçim, Çaprazlama ve Mutasyondur. Bunlar sırası ile anlatılacaktır.

2.3.1.5.1. Seçim

Seçim operatörü bazı kaynaklarda tekrar üreme veya yeniden kopyalama olarak da anılabilmektedir. Seçim operatörü, çözüm topluluğu içerisinden uygun olan bireylerin seçilmesi ve bunların sonraki topluluğa kopyalanarak hayatta kalmalarıyla ilgilidir. Seçim işleminde bireyler, uygunluk fonksiyonlarına göre kopya edilirler. Çözüm uzayındaki her bir bireyin uygunluğu baz alınarak ne sayıda kopyasının olacağına karar verilir. Oransal olarak en iyi bireylerden daha fazla kopya alınırken en kötü bireylerden kopya alınma olasılığı düşüktür. Bu, hayatta kalmak için uygunluk stratejisinin GA’ya sağladığı avantajdır.

Seçim işlemini gerçekleştirmede yaygın olarak kullanılan birkaç farklı yöntem vardır. Bunlar sırası ile anlatılacaktır.

2.3.1.5.2. Rulet Tekeri Seçimi (Roulette Wheel Selection)

Seçilen bireyler arasındaki uygunluk değerlerini birbirlerine yakın (baskın bireyler) oluşturmamak için genellikle rulet tekeri seçim yöntemi kullanılır. Bu yöntemde, bireylerin uygunluk değerlerine paralel olarak rulet tekeri üzerinde kapladıkları alan artar. Tekerleğin rasgele döndürülmesinden sonra, bireyin bir sonraki

nesil için seçilmesi, tekerlek üzerinde kapladığı alanla doğrudan bağlantılıdır. Bu yöntem, düşük uygunluğa sahip bireylere de seçilme hakkı verir.

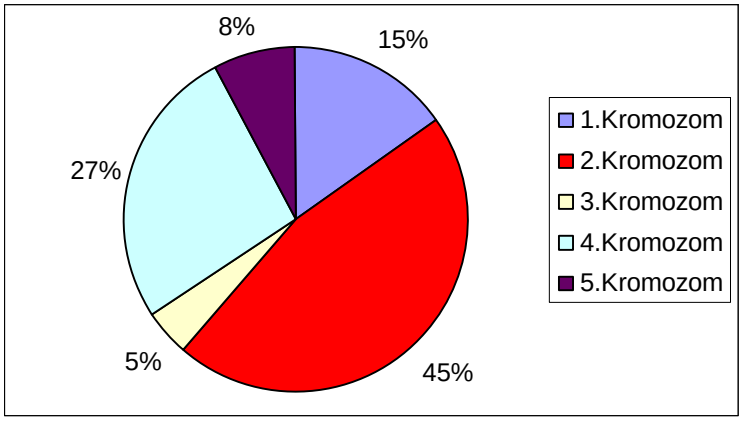
Rulet tekeri seçim algoritması aşağıdaki gibidir (NABİYEV, 2003):

1) Çözüm grubundaki bütün kromozomların uygunluk değeri hesaplanır ve bu değerler toplanır,
2) Her bir uygunluk değeri, uygunluk değerleri toplamına bölünerek bireylerin [0,1] aralığında seçilme olasılıkları belirlenir,
3) (0, toplam uygunluk değeri) sayı aralığında rasgele bir sayı seç,
4) Seçilen sayıyı kromozomların uygunluk değerleriyle karşılaştır, eğer kromozomun uygunluk değeri rasgele seçilen sayıdan büyükse o kromozomu seç.
5) Yeni jenerasyondaki kromozom sayısı eski jenerasyona ulaşıncaya kadar 3. adımdan devam edilir.

Çizelge 2.4. Örnek kromozomlar ve uygunluk değeri yüzdeleri

	Uygunluk	%
1.Kromozom	209	15
2.Kromozom	621	45
3.Kromozom	64	5
4.Kromozom	361	27
5.Kromozom	106	8
Toplam	1361	100

Çizelge 2.4’ de 5 kromozoma ait uygunluk değerleri ve uygunluk değerlerinin toplamı görülmektedir. Bu toplama göre her kromozomun %’lik değerleri hesaplanmıştır. Şekil 2.7’ de beş kromozomun seçilme olasılıkları yüzde olarak verilmiştir. Rulet tekerinde seçilme ihtimali en yüksek olan kromozom 2. kromozom olmasına rağmen bu kromozom seçilmeyebilir ve seçilme olasılığı en düşük olan 3. ve 5. kromozomlar seçilebilirler.



Şekil 2.7. Rulet tekeri modeli

Rulet tekeri seçimi, eğer uygunluk bakımından baskın bireyler çoğunlukta ise sorun çıkartabilir. Örneğin en iyi kromozomun uygunluğu tüm rulet tekerinin %95'i ise diğer kromozomların seçilme şansı azalacaktır. Bunu önlemek için sıralama seçimi kullanılabilir.

2.3.1.5.2.1 Sıralama Seçimi (Rank Selection)

Sıralama seçimi önce populasyonu sıralar ve daha sonra her kromozom, uygunluğu bu sıralamadan sonra alır. En kötüsü 1 uygunluğunu alacak, ikinci en kötü 2 ve en iyisi N uygunluk değerini alacak ki N de populasyondaki kromozom sayısıdır (NABİYEV, 2003). Bundan sonra her kromozomun seçilme hakkı olacaktır. Bu yöntemde en iyi kromozomların seçilme olasılığı diğerlerinden fazla değildir.

2.3.1.5.2.2 Sabit Durum Seçimi (Steady-State Selection)

Bu yöntem, yerine geçme yöntemi olarak da adlandırılabilir. Bu seçimin ana fikri toplumun var olan kromozomlarının büyük bir kısmının yeni nesle aktarılmasıdır.

Sabit durum seçimi şu şekilde çalışmaktadır. Her yeni nesilde yüksek uygunluk değerine sahip kromozomlar yeni yavruları oluşturmak için seçilir ve düşük uygunluk değerine sahip yavrular kaldırılarak yerlerine bu yeni oluşturulan yavrular koyulur. Toplumun geri kalan kısmı aynen yeni nesle aktarılır (KALAYCI, 2006). Yani kısaca bu yöntemde alt populasyon oluşturulduktan sonra uygunluklar hesaplanır, en kötü kromozomlar, yerlerini başlangıç populasyonundaki en iyi kromozomlara bırakırlar.

2.3.1.5.2.3 Turnuva Seçimi (Tournament Selection)

Turnuva seçiminde kromozomların özellikleri karşılaştırılır ve en iyi kromozom seçilir. Birbiriyle yarıştırılan kromozom sayısı azaldıkça GA daha verimli çalışır. En iyi sonucu elde etmek için kromozomlar birbirleriyle n sayıda (n=2,3,4……) yarıştırılırlar, n sayısı arttıkça çözümdeki çeşitlilik azalır. Bu durumun oluşmaması için n sayısı genellikle 2 seçilir.

Basitçe çalışma sistemi 3 adımdan oluşur.

1) Rasgele n tane kromozom seçilir.
2) Seçilen bu n kromozomdan en iyi uygunluk fonksiyonuna sahip kromozom seçilir.
3) 1. ve 2. adımlar yeni jenerasyondaki kromozom sayısı eski jenerasyona ulaşıncaya kadar devam edilir (KERT, 2006).

Yeni oluşan jenerasyonun içinde aynı kromozomdan birden fazla gelebilir. Turnuva metodunda uygunluk değeri yüksek olan kromozom seçilir.

2.3.1.5.3. Çaprazlama

Amaç, ana kromozom genlerinin rasgele seçilecek bir çaprazlama noktasından itibaren yerini değiştirerek yavru kromozomlar üretmek ve böylece uygunluk değeri belli bir seviyede olan ana kromozomlardan, uygunluk değeri daha yüksek olan yavru kromozomlar elde etmektir. Burada önemli olan husus, çaprazlama sonrasında elde edilecek yavru kromozomların uygunluk değerlerinin ne olacağıdır. Bu işlem yapılırken amaç, ana bireylerden yavru bireylere çözümün güçlü yanlarının geçmesidir. Ancak bu durum her zaman gerçekleşmeyebilir. Bu yüzden gelişigüzel yapılan çaprazlamada, sonucun mükemmelliğe doğru gitmesi için bazen ana kromozomlardan, uygunluk değeri yavru kromozomlara göre daha yüksek olan kromozom yada kromozomlar, yeni nesilde yerlerini yavru kromozomlara bırakmayabilirler (KARABOĞA, 2004).

Çizelge 2.5’ de ikili kodlama yöntemiyle kodlanmış iki ana kromozomun çaprazlanması sonucunda elde edilen iki yavru kromozomun yapısı verilmiştir.

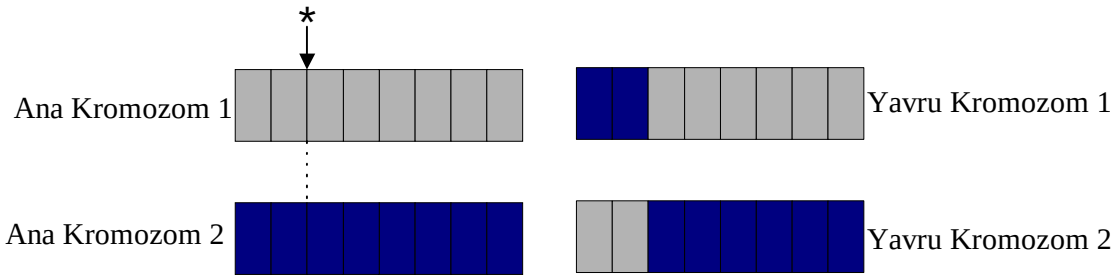
Çizelge 2.5. İki bitlik çaprazlama örneği

Ana kromozom 1	100110|00	Yavru kromozom 1	10011**01**
Ana kromozom 2	**110111|01**	Yavru kromozom 2	**11011**00

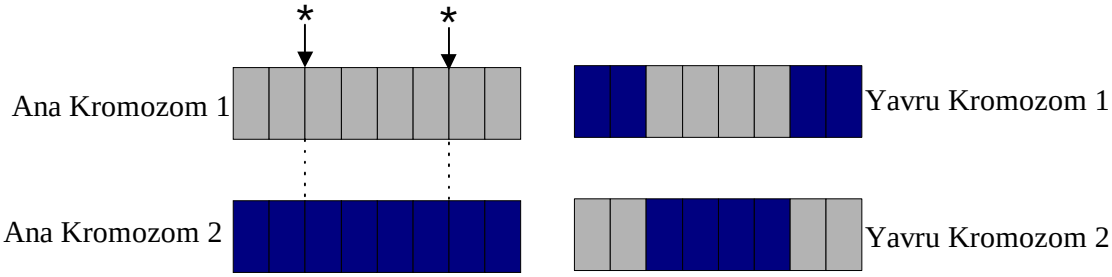
Çaprazlamadan başka “ters çevirme (inversion)” denilen bir üreme yöntemi daha vardır. Holland bunu tanımlayarak kromozom uzunluğu çok olan bireylerde çaprazlama yerine bunun kullanılmasını performans açısından önermiştir. Ters çevirme, bir kromozomu oluşturan genlerden ardışık bir grubun kendi içerisinde birbirleriyle yer değiştirerek ters dizilmeleridir. Örneğin:0111**1**01**0**1 kromozomu (her genin bir bit olduğu varsayımı ile) 5. ve 8. Gen kromozomları kendi aralarında yer değiştirdiğinde ortaya 0111**0**10**1**1 kromozomu çıkar.

Ters çevirme, genellikle kromozom uzunluğu fazla olan populasyonlara uygulanır.

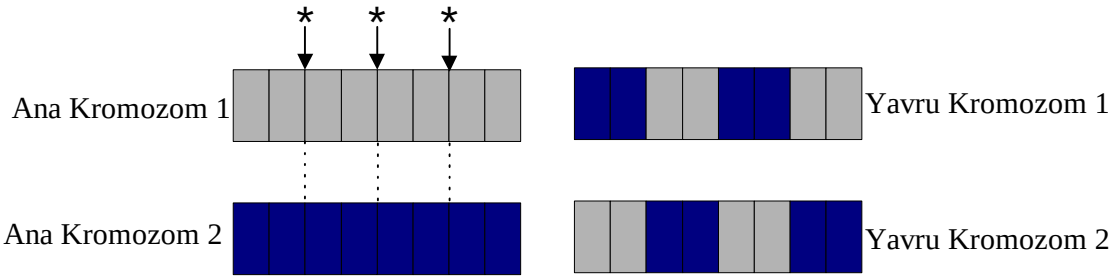
Çaprazlama işlemi, rasgele seçilecek bir yada daha fazla çaprazlama noktasından gerçekleştirilebilmektedir. Bu durumlar çaprazlama noktaları yıldız işaretiyle belirtilmek suretiyle Şekil 2.8, Şekil 2.9 ve Şekil 2.10’ da görülmektedir.



Şekil 2.8. Tek noktalı çaprazlama



Şekil 2.9. İki noktalı çaprazlama



Şekil 2.10. Üç noktalı çaprazlama

Çaprazlama oranı, yavru bireylerin üretilmesinde ana bireylere uygulanacak çaprazlama operatörünün frekansını belirlemede kullanılan bir parametredir. Düşük çaprazlama oranı, yeni kuşağa çok az sayıda yeni yapının girmesine neden olmaktadır. Dolayısıyla çaprazlama operatörü, algoritmada aşırı etkili operatör olarak haline gelmekte ve çözüm uzayının çok hızlı bir şekilde araştırılmasına neden olmaktadır. Ancak oran aşırı yüksekse, çaprazlama operatörü benzer veya daha iyi yapıları üretemeden kuvvetli olan yapılar çok hızlı olarak bozulduğundan algoritmanın performansı düşmektedir (KARABOĞA, 2004).

2.3.1.5.4. Mutasyon

Amaç, varolan bir kromozomun genlerinin rasgele bir ya da birkaçının yerlerini yada genin içeriğini değiştirerek yeni kromozom oluşturmaktır. Sürekli olarak yeni nesil üretimi sonucunda belirli bir süre sonra nesildeki kromozomlar birbirlerini tekrarlama konumuna gelebilir ve bunun sonucunda farklı kromozom üretimi durabilir veya çok azalabilir. İşte bu nedenle nesildeki kromozomların çeşitliğini artırmak için rasgele

seçilecek kromozomlardan bazıları mutasyona uğratılır. Mutasyon populasyonlarda çok önemlidir. Mutasyon ile gelebilecek bir değişiklik, kromozomu aranan çözüm olarak elde edilmesini sağlayabilir. Oysaki bu durumu çaprazlama sağlamayabilir. Bu durumda o gen için mutasyon kaçınılmazdır. Bu işlem çaprazlamadan sonra gelir.

Mutasyonun yapılıp yapılmayacağını bir olasılık testi belirleyebileceği gibi mutasyon işlemi her yeni nesle belli bir oranda uygulanabilir. Bunların dışında adaptif mutasyon uygulaması da mevcuttur. Belli bir jenerasyon sonrasında üst üste gelen aynı uygunluk değerleri karşısında çözüm topluluğuna zenginlik katmak amacıyla mutasyon oranı arttırılıp azaltılarak çözüm uygunluklarına bağlı olarak adaptif bir mutasyon oranı uygulanabilir.

Mutasyon işlemi, ikili kodlamaya sahip bir kromozomda rasgele seçilmiş bir bit için 0'ı 1 ve 1'i 0 yapma ile gerçekleşir.

Çizelge 2.6. bit bazında hazırlanmış bir mutasyon operatörünü göstermektedir.

Çizelge 2.6. Mutasyon operatörü

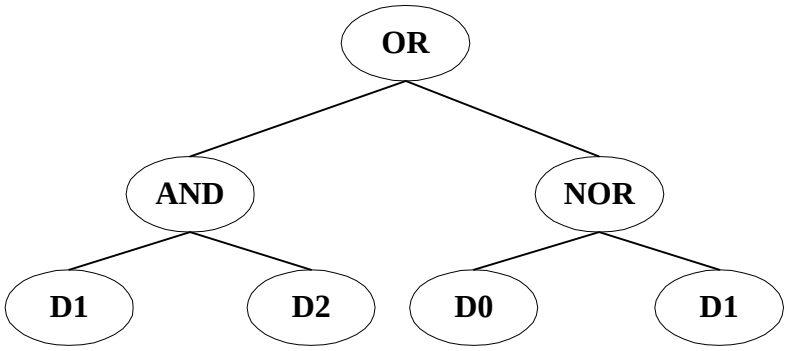
Mutasyon öncesi kromozom	110**1**111000011110
Mutasyon sonrası kromozom	110**0**111000011110

Mutasyon oranı, mutasyon operatörünün frekansını ifade etmektedir. Etkili bir GA tasarlamak için mutasyon oranı çok iyi kontrol edilmelidir. Mutasyon operatörü çözüm uzayında yeni bölgelere girilmesini sağlar. Yüksek mutasyon oranı, araştırmaya aşırı bir rasgelelik kazandıracak ve çözüm topluluğunun gelişmesine değil tahribatına sebep olacaktır. Çok düşük mutasyon oranının kullanılması, araştırma uzayının tamamen araştırılmasını engelleyecektir. Dolayısıyla, algoritmanın alt optimum çözüm bulmasına sebep olacaktır.

2.3.2. Genetik Programlama

Genetik Programlama metodunda, kromozomlar ağaç dallarına benzer bir şekilde kodlanırlar yani daha önce de bahsedildiği gibi ağaç kodlama ile kodlanırlar. Bu dalların kesişim noktaları fonksiyonlardan oluşurken yapraklar ise sabit sayı veya

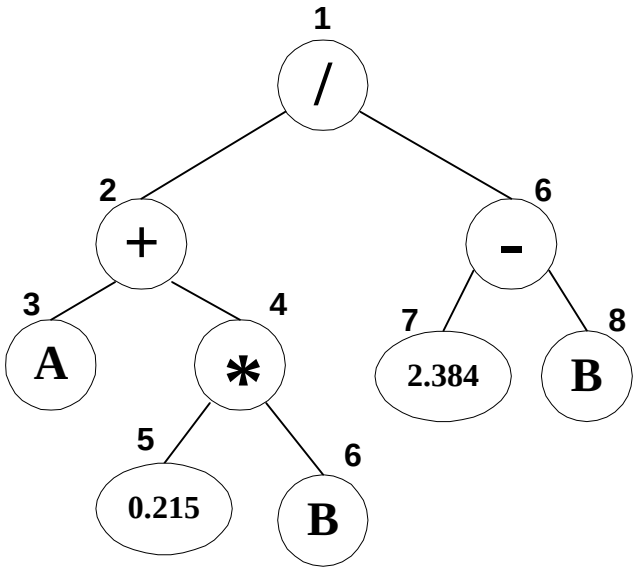
değişkenler ile fonksiyonlardan oluşan terminallerdir. Şekil 2.11' de ağaç kodlamalı bir kromozoma ait yapı yer almaktadır.



Şekil 2.11. GP'de kullanılan lojik ifade içerikli kromozom örneği

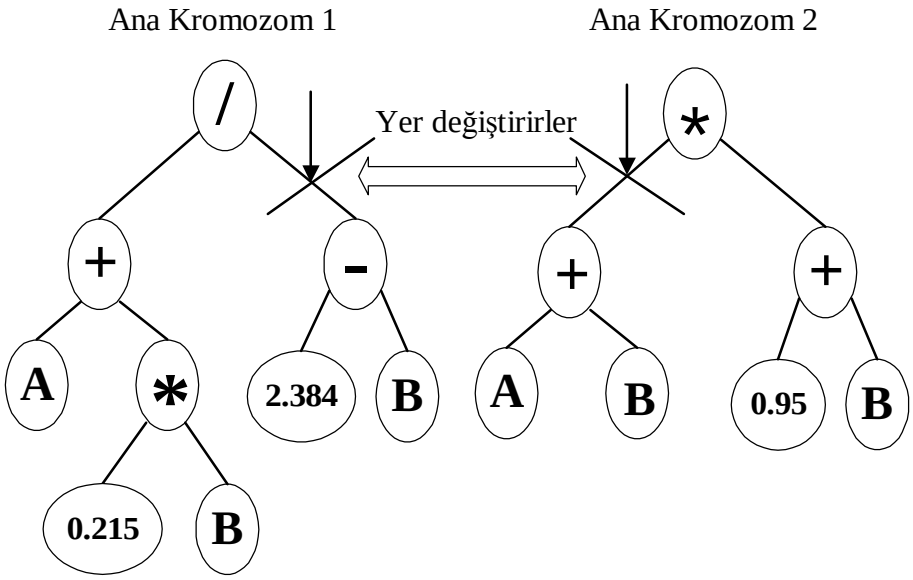
Şekil 2.11' de yer alan kromozom D0, D1, ve D2 gibi 3 farklı girişe sahip (D1 AND D2) OR (D0 NOR D1) *boolean* eşitliğini ifade etmektedir.

GP de kromozomlar sadece lojik işlemler için kullanılmaz aynı zamanda matematiksel fonksiyonları da ifade edebilirler. Örneğin, (A+0215B) / (2,384-B) gibi bir matematiksel eşitliğin GP de kromozom olarak kodlanmış hali Şekil 2.12' de yer almaktadır.

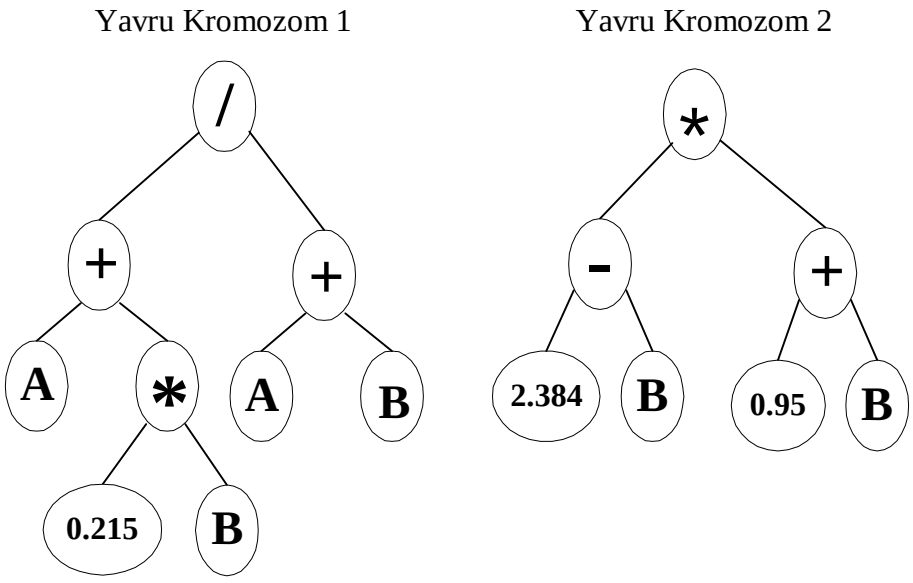


Şekil 2.12. Matematiksel fonksiyonlarla oluşturulmuş kromozom yapısı

GP de kromozomların çaprazlanması işlemi, her iki kromozomunda alt dallarının belli noktalardan kırılıp birbirleri arasında değiştirilerek bu yeni dalların kaynaştırılması şeklinde gerçekleşmektedir. Ok işaretleri ile gösterilen noktalardan gerçekleştirilecek çaprazlama işleminin öncesi ve sonrasındaki kromozom yapıları Şekil 2.13 ve Şekil 2.14' de olduğu gibidir.

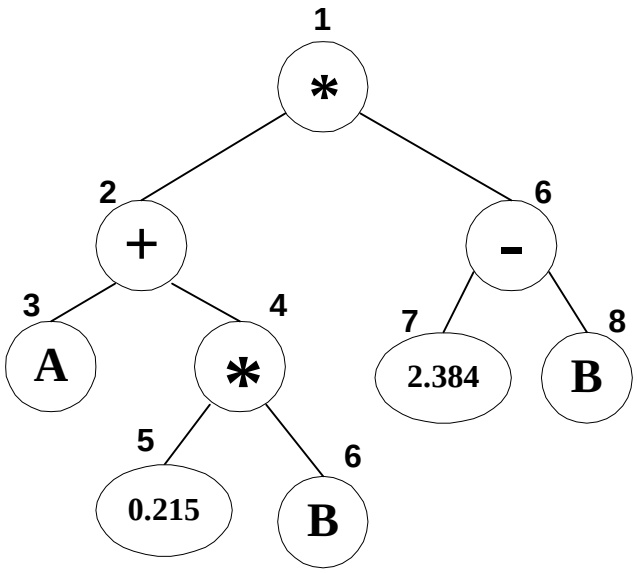


Şekil 2.13. Çaprazlama öncesi ana kromozomlar



Şekil 2.14. Çaprazlama sonrası oluşan yavru kromozomlar

GP'de mutasyon işlemi, fonksiyonun farklı bir fonksiyon ile değiştirilmesi yada iki fonksiyonun yer değiştirmesi şeklinde gerçekleşebilir. Bu durum işlem gören sabit sayı veya değişken için de aynıdır. Örneğin Şekil 2.12' de görünen kromozomun 1 numaralı bölme fonksiyonuna mutasyon uygulandığına 1 numaralı fonksiyonu çarpma olan Şekil 2.15' de görünen kromozom elde edilebilir.



Şekil 2.15. Mutasyona sonrası kromozom

Bu mutasyon işlemi farklı bir fonksiyona uygulanabileceği gibi herhangi bir sabit sayı veya değişkenin içeriğini değiştirme şeklinde de uygulanabilir.

2.3.3. Evrimsel Programlama

Evrimsel Programlama (EP) yaklaşımının GA'dan temel faklılığı, kromozom yapılarının oluşturulması ve genetik operatörlerdeki kullanımıdır. GA'daki kromozomların oluşturulmasında ikili sayı sistemi EP'de yerini tam veya reel sayılara bırakır. Ayrıca EP, probleme özgü bilgi katan özel genetik operatörler kullanır. Bu özel genetik operatörlerden dolayı bu yaklaşım, daha kısa bir kromozom yapısına ve daha hızlı işleyen bir evrimsel sisteme sebep olur.

Yukarıda bahsedilen GA, GP ve EP dışında pek de yaygın olarak kullanılmayan VGAs (Variable Length Chromosomes Genetic Algorithms) ve PGAs (Production Genetic Algorithms) gibi alternatif EH yaklaşımları vardır.

2.4. Hedef Uygulama Alanları

EHW'nin uygulama alanları aşağıda belirtildiği gibi sınıflandırılabilir.

1. Analog ve Dijital Devre Tasarımları (Analog and Digital Circuit Design)
2. Robotik ve Kontrol (Control and Robotics)
3. Desen tanıma (Pattern Recognation)
4. Hata tolerans (Fault-Tolerant)
5. VLSI (Very Large Scale Integration)

Bu sınıflandırma, literatürde belirtilen ve halen çalışılmakta olan uygulamalar baz alınarak yapılmıştır. Önümüzdeki yıllarda bu çalışmaların çeşitliliğinin artması beklenmektedir.

2.4.1. Devre Tasarımı

Analog ve dijital elektronik devre tasarımı EE'nin öncelikli uygulamasıdır. Harici EHW için geniş ölçekli devre simulatörlerinin varlığı, dahili EHW sistemlerinde kullanılan FPGAs (Field Programmable Gate Arrays) gibi programlanabilen donanımlar araştırmacıların EHW sistemleri üzerinde çalışmalarına olanak sağlamıştır.

2.4.2. Robotik ve Kontrol

FSM (Finite State Machine) ve diğer basit modeller gibi evrimleşebilen kontrol uygulamalarındaki alt sistem sayısı sağlanırsa, bu alan EHW'nin en gelecek vaat eden alanlarından biridir. Robotların kontrol sistemlerini geliştirmek için evrimsel tekniklerin kullanıldığı "Evrimsel Robotik (Evolutionary Robotics)" alanındaki çalışmalar yoğun bir şekilde sürdürülmektedir. Bunlara ilerleyen kısımlarda değinilecektir.

2.4.3. Desen Tanıma

Yapay Sinir Ağları (YSA), desen tanıma uygulamalarında giderek artan bir şekilde kullanılmıştır. Higuchi ve arkadaşları sinir ağları hakkında EHW'nin basit avantajlarını hedefleyip EHW'yi bu alanda da kullanmaya çabaladılar. Higuchi tarafından yapılan EHW ile YSA arasındaki karşılaştırma tablosu aşağıdaki gibidir.

Çizelge 2.7. EHW ile ANN karşılaştırma tablosu

Kriter	**EHW**	**YSA**
Hız (Speed)	Daha hızlı	Daha yavaş
Adaptiflik (Adaptiveness)	On-line	Off-line
Tasarımcı bilgisi (Designer Knowledge)	Az gerekli	Az gerekli
Donanım dönüşümü (Hardware Conversion)	Direkt	Daha karmaşık

Higuchi tarafından yapılan bu karşılaştırma çalışması incelendiğinde EHW'nin YSA'ya olan üstünlüğü hız, adaptiflik ve donanım dönüşümü kriterleri açısından açıktır.

2.4.4. Hata Tolerans Sistemleri

Hata-Tolerans, donanım sistemleri meydana getirmenin en yaygın yolu olup, ekonomik ve bazen de uygulanabilir bir çözüm olmayan baştan sona kritik elemanların kopyalanmasıdır. Adaptif sistem kavramının kullanımı hata-tolerans sistemine çok daha avantajlı bir çözüm olarak düşünülebilir.

İdeal olarak adaptif sistemler, arıza oluşumu gibi ortam değişimlerine karşı kendi donanım yapılarını değiştirebilirler. Örnek olarak ideal adaptif bir bilgisayar gerektiğinde her bir saat darbesi geçişinde kendi yapısını özellik olarak değiştirebilmelidir.

EHW, baştan sona genetik öğrenme ile tekrar ayarlanabilen bir yapının sentezi olduğundan, hata-tolerans sistemleri için ideal bir yaklaşımdır. Bu durumda genetik öğrenme işlemi, verilen özel ortam koşullarına en uygun donanım yapısını arayan bir yapı oluşturur. Higuchi'nin bahsettiği gibi, sistemi gerçek zamanlı uygulamalara uygun

hale getirmek için öğrenilmiş sonuç doğrudan yeni bir donanım sistemine dönüştürülebilir.

2.4.5. VLSI

Her ne kadar bu noktada VLSI çiplerinin tasarımında EHW uygulamalarına yönelik birkaç çalışma rapor edilmişse de bu, evrimsel sistemlerin gelecekteki uygulamalarının tartışılmaz bir alanıdır. VLSI çip tasarımında aşağıda belirtilen problemler EHW yaklaşımına potansiyel birer adaydır.

1.Analog ve Dijital Devre Düzeni (Circuit Layout (analog and digital))

2.Yerleşim (Placement)

3.Çizim (Routing)

Devre düzeninde evrimleşecek yapılar, transistörler, geçiş kapıları, katmanlar veya bir kütüphanenin hücreleri bile olabilir.

Yerleşim ve Çizim basitçe EH yaklaşımlardan büyük ölçüde fayda sağlayabilen tümleşik optimizasyon problemidir.

2.5. Evrimleşme Platformu

Evrimleşme platformu, evrim geçiren devrenin yer alacağı donanım ile ilgilidir. Evrimleşme platformlarını üç ana sınıfa ayırabiliriz;

1. Tümleşik Devre Düzeni

2. Amaca Yönelik Tahsis Edilmiş Donanım

3. Programlanabilen Tümleşik Devreler

Bunlara ait detaylar aşağıdaki kısımlarda yer almaktadır.

2.5.1. Tümleşik Devre Düzeni

Bu metotta komple devre düzeni elde etmek için metal oksit ve silikon gibi tümleşik devre katmanlarının yanı sıra transistör veya geçiş kapıları gibi komple donanımlar evrimsel işleyişle yürütülür. VLSI tasarımı bu yaklaşıma bir örnektir. Bu durumda çok sayıda olası devreyi değerlendirmek için haricen yapılan (extrinsic) tasarım yaklaşımı geçerli tek plandır. Gelecekte EHW yaklaşımının kullanacağı

elemanların “Optik (optical)” ve “Galyum Arsenür (Galium Arsenide)” olması beklenmektedir (YAO ve HIGUCHI, 1999).

2.5.2. Amaca Yönelik Tahsis Edilmiş Donanım

Amaca yönelik tahsis edilmiş donanım genel olarak ASIC (Application-Specific Integrated Circuit) olarak adlandırılır. Bir ASIC, genel amaçlı kullanımlardan ziyade özel amaçlı kullanımlara hitap etmektedir. Örneğin sadece bir cep telefonunu çalıştırmak için kullanılan çip bir ASIC tir. Buna karşın çoğu farklı uygulamaları gerçekleştirmek için birbirlerine bağlı tümleşik vaziyette bulunan 74XX ve 4000 serilerinden bahsedilebilir (ANONYMOUS, 2006).

Yıllar geçtikçe boyutlar küçüldü. Maksimum içeriğe sahip çipte kapı sayısı 5000 den 100 milyonun üzerine çıkarıldı. Modern ASIC’ler; RAM, ROM, EEPROM, Flash ve diğer yapıları barındıran hafıza blokları ve 32 bitlik işlemciler içerir. Böylesi bir ASIC, SoC (System-on-a-chip) olarak adlandırılır. Dijital ASIC tasarımcıları, ASIC fonksiyonlarını açıklamak için VHDL veya Verilog gibi HDL (Hardware Description Language) kullanırlar (ANONYMOUS, 2006).

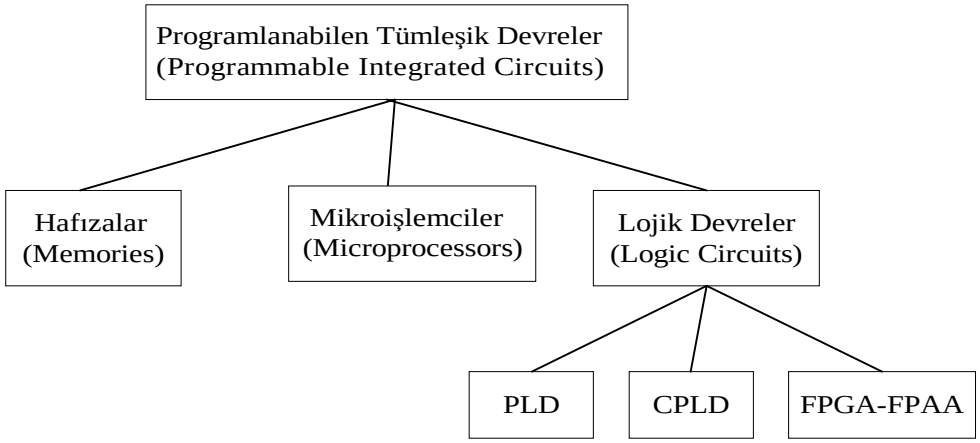
FPGA, 74XX serisinin günümüz modern lojik eşdeğeridir ve programlanabilen lojik bloklar ile aynı FPGA’nın farklı uygulamalarda kullanılmasına izin veren programlanabilir bağlantılara sahip bir breadboard gibidir. FPGA lar küçük tasarımlarda ASIC lere oranla daha efektif olabilmektedirler (WOO ve ark.,1999).

Evrimsel işleyiş, bir amaca yönelik yapı elde etmek için yönlendirilebilir. YSA elde etmeğe yönelik bir yapı, örneğin evrimsel işleyişin basit bir elemanı olarak “yapay nöronlar (artificial neurons)” kullanılarak evrimleşebilir. “Fuzzy çipler (Fuzzy chips)”, EH yaklaşımına üye olan diğer bir amaca yönelik gerçekleştirilen yapıdır. Dahili ve harici EHW yaklaşımlarının her ikisi de kullanılabilir (MANARESI ve ark.,1996).

2.5.3. Programlanabilen Tümleşik Devreler

EHW de kullanılan standart platformlar programlanabilen tümleşik devrelerden ibarettir. Bu devreler 3 kategoriye ayrılabilir : Hafızalar (Memories), Mikroişlemciler (Microprocessors) ve Lojik Devreler (Logic Circuits). EHW, esasen Programlanabilen Lojik Devreler üzerine odaklanır. Lojik devreler, 3 alt kategoriye bölünür. Bunlar sırasıyla PLD (Programmable Logic Device), CPLD (Copmlex Programmable Logic

Device) ve FPGA (Field Programmable Gate Arrays). Sınıflandırma, aşağıdaki Şekil 2.16' da gösterilmektedir.



Şekil 2.16. Tekrar ayarlanabilen devreler ailesi

Tekrar ayarlanabilen devreler ailesine sırası ile değinilecektir.

2.5.3.1. PLD

Karmaşık fonksiyonları yerine getirmek için programlanabilen tümleşik bir devredir. AND ve OR kapı dizilerinden meydana gelir. Sistem tasarımcısı lojik tasarım gerçekleştirmeden önce PLD içerisinde yer alan kapıların işlev gösterebilmesi için programlayıcı cihaz yardımıyla PLD üzerindeki sigortaların patlatılması gereklidir.

Programlanabilme derecelerine göre PROM (Programmable Read-Only Memory), PAL (Programmable Array Logic) veya PLA (Programmable Logic Arrays) olarak sınıflandırılabilir.

PLD tipleri aşağıdaki gibi sınıflandırılabilir;

- *PROM (Programmable Read Only Memory)*

Küçük tasarımlar için düşük maliyet ve yüksek hız sunar.

· *PLA (Programmable Logic Array)*

Daha karmaşık tasarımlar için esnek özellik sunar. Programlanabilme özelliği bütün PLA'larda alan etkili (field-programmable) değildir. PLA ların çoğu ROM larda olduğu gibi üretim aşamasında maskeleme yöntemiyle programlanırlar.

- *PAL/GAL (Programmable Array Logic/Generic Array Logic)*

PLA'lardan daha ucuz ve daha hızlı olup tasarımcıya iyi bir esneklik sunar.

PAL (Programmable Array Logic) terimi 1978 ortalarında *MMI (Monolithic Memories Inc.)* tarafından bulunan dijital devrelerdeki lojik fonksiyonları yerine getirmede kullanılmış programlanabilen lojik donanım (programmable logic device) ailesini tanımlamak için kullanılmıştır. PAL donanımları küçük bir PROM (programmable read-only memory) çekirdeği ve ilaveten birkaç eleman ile arzu edilen lojik fonksiyonları yerine getirebilecek çıkış lojiğinden ibarettir. Her PAL donanımı bir defaya mahsus programlanabilme (one-time programmable) özelliğine sahiptir.

GAL, *Lattice Semiconductor* tarafından bulunmuş olup PAL'ın yeni bir modeli olarak ortaya çıkmıştır. Tekrar programlanabilen yapısı sayesinde tasarımcıların yaptıkları tasarımlar üzerindeki değişiklikleri yapıya aktarabilmesi amaçlanmıştır.

2.5.3.2. CPLD

CPLD, Complex Programmable Logic Device açılımına sahip olup programlanabilen hücrelerin kombinasyonu olarak düşünülebilir. Yapı itibariyle FPGA ile PAL arasındadır. CPLD'nin blokları makro hücrelerden meydana gelir. Bu hücreler genellikle, "Multiplexer" lar veya hafıza elemanları ve programlanabilen hücrelerin girişlerini seçen bir ara bağlantı ağına (interconnection network) sahiptir.

PAL ile ortak özellikleri;

- Kalıcı bir hafıza konfigürasyonuna sahiptir. Çoğu FPGA'nın aksine harici bir ROM konfigürasyonuna ihtiyaç duymaz ve CPLD, sistem açılışında hemen işlev görebilir.
- En geniş kapasiteli donanımların tamamı için, harici pinlere bağlı giriş ve çıkış sinyallerine sahip çoğu lojik bloklar yol (bağlantı) bulmaya zorlanır.

FPGA ile ortak özellikleri;

- Tipik olarak binlerden on binlere kadar çok sayıda lojik kapıya sahiptirler.
- Bazı lojik hazırlıklarında makro hücreler ile karmaşık geri besleme yolları içeren "Disjunctive Normal Form" dan daha esnektir ve çoğunlukla kullanılan çeşitli fonksiyonları elde etmek için özelleştirilirler.

Geniş kapasiteli CPLD ile düşük kapasiteli FPGA arasındaki en dikkate değer fark CPLD içinde yer alan kaybolmayan hafızanın (non-volatile memory) varlığıdır. Bu özellikleri sayesinde CPLD ler modern dijital tasarımlarda sıklıkla kullanılırlar.

2.5.3.3. FPGA ve FPAA

FPGA, programlanabilen eleman ve bağlantılarından oluşan bir yarı iletken donanımdır. Programlanabilen lojik elemanlar AND, OR, XOR, NOT gibi basit lojik kapıların veya daha karmaşık "Decoder" veya basit matematik fonksiyonları gibi kombinasyonel fonksiyonların işlevselliğini arttırmak için programlanırlar. Çoğu FPGA'lar bu programlanabilen lojik elemanların (lojik bloklar) yanı sıra "Flip-Flop" lar veya daha karmaşık hafıza elemanlarını da içinde bulundururlar.

FPGA içindeki programlanabilen bağlantılar sistem tasarımcısı tarafından ihtiyaç duyulduğu şekilde bağlantı yapılmasına olanak sağlar. Bu lojik bloklar ve bağlantılar müşteri/tasarımcı tarafından üretim işlemi sonrasında (bu yüzden "field programmable" terimi kullanılır) programlanabilir. Böylece FPGA her ne mantıksal fonksiyon gerekiyorsa onu sağlamış olur.

FPGA'lar genel olarak ASIC eşitlerine oranla daha yavaştırlar ve karmaşık tasarımları ele alamazlar. Ancak daha kolay temin edilebilmesi ve tekrardan programlanabilmesi avantajları arasındadır. Satıcılar FPGA'ların, yapılan tasarımın işlenmesi sonrasında yeniden düzenlenemeyen (daha az esnek) versiyonlarını daha ucuza satabilmektedirler. Tasarımların gelişimi, FPGA'ların CPLD veya diğer alternatifleri olan sabitlenmiş benzer bir ASIC versiyonuna geçişi sağlayacaktır.

FPGA'lar, dijital EHW yaklaşımında en çok kullanılan yeniden yapılanabilen donanımlardır. Lojik hücreler ve I/O (Giriş/Çıkış) hücrelerinin bir dizisinden ibarettirler. Her bir lojik hücre tam bir fonksiyon elde edebilmek için programlanabilen "multiplexer", "demultiplexer" ve hafızadan ibarettir. XILINX ailesinin FPGA ları EHW araştırmacıları tarafından yoğun bir şeklide kullanılmaktadır. Her bir lojik hücre veya bir XILINX FPGA'nın fonksiyonel bloğu bir RAM veya kombinasyonel lojik devrenin doğruluk tablosunu programladığı taramalı tablo (look-up table) dan ibarettir. Ara bağlantılar (interconnections), bu FPGA ailesinin programlanabildiği anlamına gelen bir RAM içeriği tarafından kontrol edilir.

Evrimleşme platformu olarak FPGA'ların yerine hafıza kullanımı da mümkündür. Bu durumda, hafıza içeriği genetik olarak programlanacaktır.

FPGA'ların yanı sıra bir de FPAA (Field Programmable Analog Array) mevcuttur. FPAA, FPGA'nın analog eşitidir. FPGA'daki çok sayıdaki programlanabilen dijital donanımın aksine, az sayıda "ayarlanabilen analog bloklar (Configurable Analog Blocks)" dan oluşmaktadır. FPAA, "programlanabilen kondansatör dizisi (Programmable Capasitor Arrays)", "programlanabilen direnç dizisi (programmable resistor arrays)" ve bir işlemsel yükselteçten oluşan bloklardan meydana gelmektedir.

3. ÖNCEKİ ÇALIŞMALAR

EHW araştırmaları hızlı bir şekilde dallanmaktadır. Bu yüzden EHW alanında yapılan çalışmaları daha iyi anlayabilmek için yapılacak sınıflandırma yararlı olacaktır. TORRESEN (2006) yaptığı bu sınıflandırmada aşağıda belirtilen notasyonları kullanılmıştır.

Evrimsel Hesaplama (Evolutionary Computation-EC);

Kullanılan EH metoduna göre ikiye ayrılır.

- *Genetik Algoritma (GA)*
- *Genetik Programlama (GP)*

Teknoloji (Technology-TE);

Hedeflenen EHW alanının teknolojisi açısından ikiye ayrılır.

- Dijital (D)
- Analog (A)

Yapı Blokları (Building Blocks-BB);

Hedeflenen sistemi oluşturan elemanlar açısından bakıldığında üçe ayrılır.

- Analog elemanlarla oluşturulan EHW: Transistör, direnç, bobin, kondansatör gibi elemanlar
- Lojik kapı seviyesinde oluşturulan EHW: OR, AND, NOT gibi lojik kapılar
- Fonksiyon seviyesine oluşturulan EHW: Sinüs jeneratörler, toplayıcılar, multiplexerlar gibi fonksiyonlar

Hedef Donanım (Target Hardware-THW);

EH'lerle elde edilen sonuçların uygulandığı donanımlar açısından iki ana gruba ayrılır.

- Mevcut Ticari Donanımlar: Çoğunlukla FPGA'lar kullanılır. FPGA'lar bir bit dizisinin girilmesi ile birbirlerine bağlanan düzenlenebilir dijital

kapılardan oluşurlar. Girilen bu bit dizisi kapıların birbirlerine nasıl bağlanacaklarını belirler. FPGA'ların yerine FPAA'lar da kullanılabilir. FPGA'lardaki programlama prensibi ile çalışırlar ancak FPGA'lardaki dijital kapıların yerini analog elemanlar alır.

- Özellikleri kullanıcı tarafından belirlenen donanımlar: ASIC ler tamamen kullanıcı tarafından tasarlanan çiplerdir.

Uygunluk Hesaplama (Fitness Computation-FC);

Uygunluk hesaplama yöntemine göre EHW ikiye ayrılır.

- Harici EHW (EXTrinsic): Evrimden kaynaklanan değişim işlemi yazılımda gerçekleşir ve sadece en seçkin kromozom donanıma yazılır. Offline EHW olarak da anılır.
- Dahili EHW (INTrinsic): Donanım, her jenerasyonun her kromozomu için yeniden düzenlenir (şekillenir). Online EHW olarak da anılır.

Evrim (Evolution-EV);

Donanımda üstlenilen evrim derecesine göre üçe ayrılır.

- Çip dışı evrim (Off-chip Evolution): EH, işlemcisi ayrılmış bir ortamda gerçekleştirilir.
- Çip içi evrim (On-chip Evolution): EH, hedef EHW'yi içeren çiple ortaklaşa çalışan, işlemcisi ayrılmış bir ortamda gerçekleştirilir.
- Komple Donanım Evrimi: EH özel bir donanım içerisinde gerçekleştirilir.Örneğin bir işlemci ile çalışmadan

Kapsam (Scope-S);

Evrimin kapsamına göre ikiye ayrılır.

- Statik Evrim (Static Evolution-S): Devre normal işleme katılmadan önce evrim sonlanır. Normal işlem sırasında evrim uygulanmaz. Evrim, devre optimize aracı olarak kullanılır.
- Dinamik Evrim (Dynamic Evolution-D): Devre işlem aşamasındayken evrim gerçekleşir. Bu da devrenin online adapte olmasını sağlar.

Bu değinilen notasyonlar doğrultusunda EHW alanında yapılan çalışmaları Çizelge 3.1' de görüldüğü gibi özetlemek mümkündür.

Çizelge 3.1. Uygulama alanları ve detayları

Uygulama	**EC**	**TE**	**BB**	**THW**	**FC**	**EV**	**SC**
Adaptif Equalizer	GA	D	Neuron	Özel	INT	On-chip	S
Yükselteç ve Filtre Tasarımı	GA	A	T/R/L/C	Özel	EXT	Off-chip	S
Analog Devre Sentezi	GP	A	R/L/C	Özel	EXT	Off-chip	S
Karakter Tanıma	GA	D	Kapı	Ticari	EXT	Off-chip	S
Clock Ayarlama	GA	D	Kapı	Özel	INT	Off-chip	S
Dijital Filtre Tasarımı	GA	D	Kapı	-	EXT	Off-chip	S
IF Filter Ayarlaması	GA	A	Filtre	Özel	INT	Off-chip	S
Görüntü Sıkıştırma	GA	D	Piksel	Özel	EXT	On-chip	D
Multi-spect. Image Rec.	GA	D	Fonksiyon	Ticari	EXT	Off-chip	S
Sayı Tanıma	GA	D	Kapı	Ticari	EXT	Off-chip	S
Protez El	GA	D	Kapı	Özel	INT	Complete	S
Yol Görüntüsü Tanıma	GA	D	Kapı	Ticari	EXT	Off-chip	S
Robot Kontrolü	GA	D	Kapı	Ticari	INT	Complete	D
Robot Kontrolü	GA	D	Kapı	Ticari	INT	Off-chip	S
Sonar Sınıflandırma	GA	D	Kapı	Ticari	EXT	Off-chip	S

EHW uygulamalarındaki üç evrimsel yaklaşım üzerine yapılan çalışmaların örnekleri sırasıyla belirtilmiştir.

THOMPSON (1999), iki seviyeli frekans belirleyen devre (two level frequency detecting circuit) ve bir osilatör geliştirmek için ve her iki olaydan da iyi sonuçlar elde etmek adına *Specific Adaptation Genetic Algorithm (SAGA)* kullanmıştır. Devreleri kodlamak için her biri 101 kısımdan (segments) oluşan doğrusal bit dizileri uygulayan geleneksel GA'lara benzer yapıda bir kromozom yapısı kullandı. Boolean fonksiyonu kodlu segmentleri, düğüm noktası ve düğüm noktasındaki kapının giriş kaynakları oluşturmuştur. Thompson'un kromozom yapısı için kullandığı gösterimi Çizelge 3.2' de yer almaktadır.

Çizelge 3.2. Bir düğüm için segment genotipi

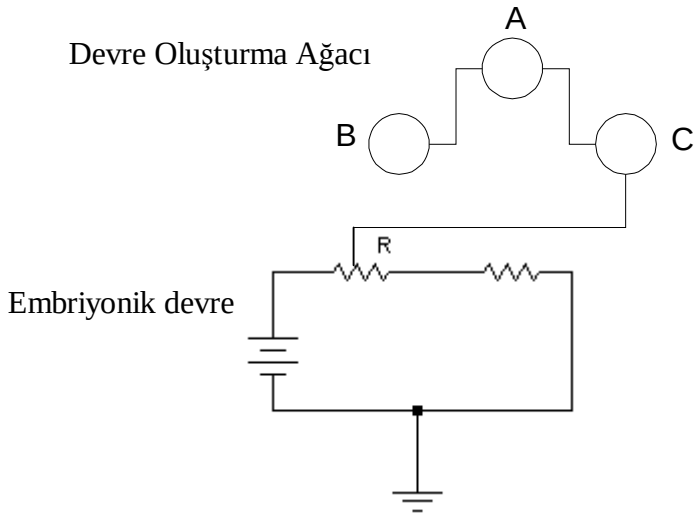
Bitler (Bits)	Anlamı (Meaning)
0-4	Kullanılmayan (Junk)
5-7	Düğüm noktası fonksiyonu (Node Function)
8-15	İlk giriş işaretçisi (pointer to first Input)
16-23	İkinci giriş işaretçisi (pointer to second input)

İkinci yaklaşım; John Holland'ın ortaya attığı GA'nın uzantısı, farklı boyut ve şekillerdeki kromozomların ağaçlar gibi simgelendiği, GP'dir. GP'de kullanılan ağaç gösterimi ve elektrik devrelerinin grafiklerle ifadesi arasındaki farklar Stanford Üniversitesinden John Koza ve arkadaşları tarafından aşağıdaki gibi yorumlandı.

1. GP kromozomları–ağaç dallarına benzer bir şekilde dallanıp etiket alırlar

2. Elektrik Devre Grafları (Electric Circuit Graphs) – Her hattın bir döngüye sahip olduğu döngüsel graflar vardır. Her hat, eleman tipi (direnç, transistör vs.) ve eğer harf ise nümerik değerini içeren iki etikete sahiptir.

Onlar, elektronik devreler tarafından belirlenen döngüsel grafları (cyclic graphs) ve GP'nin ağaçları arasında bir bağlantı kurma yolu olarak bir "devre oluşturma ağaç (circuit constructing tree)" ileri sürdüler. Devre oluşturma programının ağaçları (circuit constructing programming trees), eleman oluşturan fonksiyonları ve bağlantı yenileme fonksiyonları içerebilir. İlki, geliştirilecek devreye, kondansatörler ve transistörler gibi özel elemanları yerleştirirken, diğeri, örneğin özel elemanların paralel veya seri yorumlarını oluşturarak, geliştirilecek devre topolojisini değiştirir. Evrimsel işleyişin başlangıç devresi embriyonik elektriksel devre (embryonic electrical circuit) olarak adlandırılır. Şekil 3.1, özel bir uygulamada kullanılan embriyonik devreyi, devre oluşturma ağacı ile birlikte göstermektedir. Bu örnekte C fonksiyonu, R direnci üzerinden işlem görecektir.



Şekil 3.1. Devre oluşturma ağacı ve embriyonik elektriksel devre

Devre oluşturma ağacı kullanılarak alçak geçiren analog bir filtre geliştirildi. John R. Koza ve grubu, evrimsel işleyiş tarafından elde edilen en iyi bireylerin "Butterworth" ve "Chebychev" filtre topolojileriyle benzerlik taşıdığını gözlemlediler (KOZA, 1995).

Hitoshi Hemmi ve takipçileri de ağaç benzeri bir gösterim kullandılar. Onlar, HDL (Hardware Description Language) dilbilgisini, kromozomların kodlandığı HDL programlarına çevirmek için üretim kuralları kullandılar. Beş genetik operatör kullanıldı. Bunlar; Çaprazlama, Mutasyon, "Gen Kopyalama, Birleştirme (Fusion) ve Silme (Deletion). Bu program yaklaşımını kullanarak ardışık bir toplayıcı geliştirmede güzel sonuçlar elde ettiler (HIGUCHI ve ark., 2000).

Üçüncü yaklaşıma örnek olarak, EP'den bahsedilebilir. Hiroaki Kitano, "Grammar Encoding Method" adlı yöntemin etkinliğini göstermişlerdir. Bu metotta, grafları geliştirmek ve tamsayıları kromozomlara kodlamak için yeniden yazma kuralları (rewritting rules) kullanılmıştır. Hiroaki Kitano bu yaklaşımı kullanarak yaptığı iki deneyde altı girişli multiplexer ve çeşitli EXOR devreleri elde etmeyi başarmıştır (HIGUCHI ve ark., 1999).

GA'nın ardışık lojik devrelerde kullandığı bazı konular aşağıda tarih sırasına göre verilmiştir.

- 1993 yılında Higuchi, GA'yı "Durum Geçiş Grafları (State Transistion Graph)" tipinde dijital lojik devreler elde etmiştir (ALI ve ark. ,2004).
- 1994 yılında Hemmi, "Seri Toplayıcı (Serial Adder)" elde etmiştir (ALI ve ark. ,2004).
- 1995 yılında Thompson, DSM (Dynamic State Machine) kullanarak "Mr. Chips" olarak bilinen duvardan kaçan bir robot uygulaması gerçekleştirmiştir (THOMPSON, 1995).
- 1998 ve 1999 yıllarında Manovit ve Chongstitvatana, PLD ve GAL kullanarak senkron ardışık lojik devrelerle "Frekans detektörü (Frequency detector)", "Tek Parite Detektörü (Odd Parity Detector)", "Mod-5 Sayıcı (Module-5 Counter)", ve "Seri Toplayıcı (Serial Adder)" elde etmişlerdir (MANOVIT, 1998; CHONGSTITVATANA, 1999).
- 2000 yılında Aporntewan, Seri Toplayıcı, Mod-4 Sayıcı, "0101 Detektörü (0101 Detector)" ve "Tersinir 8 Sayıcı (Reversible 8 Counter)" elde etmiştir (APORNTEWAN, 2000).
- 2004 yılında Ali ve arkadaşları tarafından Mod-4 Sayıcı, 1010 Detektörü elde edilmiştir (ALI ve ark. ,2004).
- 2004 yılında Aksoy, D tipi Flip-Flop kullanarak "3-bit ikili sayıcı (3-bit binary counter)"elde etmiştir (AKSOY, 2004).

Bu çalışmalardan Higuchi, Hemmi, Ali ve arkadaşları ile Aksoy tarafından gerçekleştirilenler harici EHW yaklaşımını kullanırken; Thompson, Manovit, Chongstitvatana ve Aporntewan tarafından gerçekleştirilenler ise dahili EHW yaklaşımını kullanmışlardır.

4. MATERYAL VE YÖNTEM

4.1. Materyal

Yazılım, 2.6 GHz işlemci, 512 Mbyte RAM, 64 Mbyte ekran kartına sahip PC (Personel Computer) kullanılarak, Borland Delphi 7.0 programı ile geliştirilmiştir.

4.2. Yöntem

Çalışmada Borland Delphi 7.0 versiyonu kullanılarak belirlenen sayıdaki girişler ve çıkış değerleri ile arzu edilen durum geçişleri sağlanarak ardışık lojik devreler tasarlanıp optimize edilmiştir. Bu devre tasarımı ve optimizasyonu gerçekleştirilirken yöntem olarak GA kullanılmıştır.

4.2.1. Genetik Algoritma

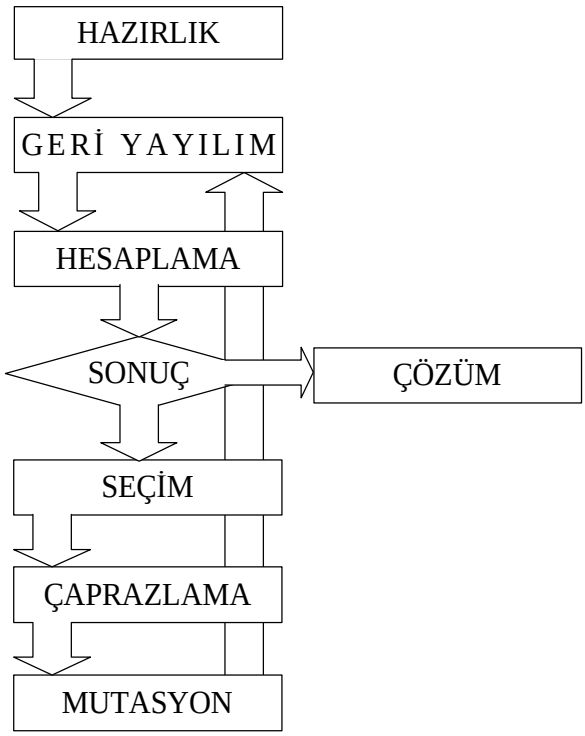
Bugün endüstrinin her alanında optimizasyona gerek duyulmaktadır. Son zamanlara kadar optimizasyon tekniklerinde matematiksel hesaplama yöntemleri kullanılmakta iken, bugün, optimizasyon tekniği olarak evrimin doğal işleyişini taklit eden EH yöntemleri vardır. Charles Darwin'den etkilenen John Holland tarafından biyolojik evrim esasına dayandırılan bu optimizasyon tekniğinde, doğal seçim ile hayatta kalan ebeveynlerin yavruları içindeki en iyi bireyin seçildiği bir yöntem izlenir. Bu yavrulardan daha sonra yeni yavrular oluşturularak, çözüm topluluğunu oluşturan bireyler daha da optimize edilir. GA da benzer şekilde en uygun çözümleri, çözüm topluluğu içerisinden seçer ve daha çeşitli çözüm elde edebilmek için çaprazlama ve mutasyon gibi genetik operatörler kullanır. Seçilen en iyi birey daha önceki jenerasyondan daha da iyi olacaktır.

GA'nın diğer optimizasyon ve arama yöntemlerinden farklılıkları şu şekilde sıralanabilir:

1) GA optimize edilecek parametrelerin kodlarıyla çalışır, parametrelerin kendilerini kullanmaz.
2) GA, arama uzayında bireylerin uygunluk değerini bulmak için sadece sonuç bilgisini (amaç fonksiyonu) kullanır, türevleri veya diğer yardımcı bilgileri kullanmaz.
3) GA, bir tek yerden değil bir grup çözüm içinden arama yapar.

4) GA ne yaptığı konusunda bilgi içermez, nasıl yaptığını bilir.
5) Olasılık kurallarına göre çalışırlar. Programın ne kadar iyi çalıştığı önceden kesin olarak belirlenemez.
6) GA, çözümlerden oluşan popülasyonu eş zamanlı incelediği için lokal maksimumlara takılmamasıdır.

Bu farklılıkları nedeniyle GA etkin arama yaparak çözüme çabuk ulaşır.



Şekil 4.1. GA'nın akış diyagramı

GA Darwin'in, en iyinin hayatta kalması fikrine dayanarak sürekli iyileşen çözümler üretir. (GOLDBERG, 1989). Bunun için "iyi" nin ne olduğunu belirleyen bir uygunluk fonksiyonu ve yeni çözümler üretmek için çaprazlama, mutasyon gibi operatörleri kullanır. GA'nın akış diyagramı Şekil 4.1' de gösterilmiştir

GA, hazırlık ile başlayan hesaplama işlemi sonrasında belirlenen hedefe ulaşıp ulaşmadığını test eder. Belirlenen hedefe ulaşılmamış ise seçim algoritmasını, çaprazlama ve mutasyon operatörlerini çalıştırır. Sonuca ulaşıncaya kadar bu işlemler devam eder.

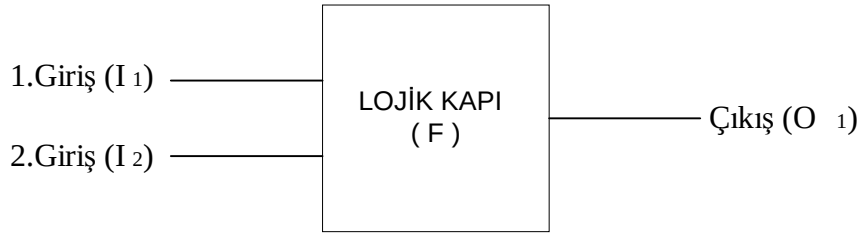
4.2.1.1. Hazırlık

Bu bölümde çözüm topluluğunu meydana getiren kromozomların oluşturulması söz konusudur. Kromozomlar ihtiyaca göre istenilen sayı tabanında, reel sayılarla veya harflerle oluşturulabilir. Problemin çözümünde doğal sayılar kullanılmıştır.

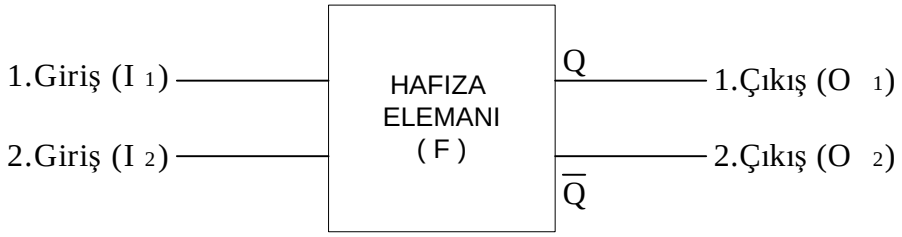
Çizelge 4.1' deki kromozom yapısında da görüldüğü üzere her bir gen 5 parçadan oluşmaktadır. Bu parçalar sırası ile 1. giriş (I_1), 2. giriş(I_2), fonksiyon (F), 1. çıkış (O_1) ve 2. çıkış (O_2) olmaktadır. Bu duruma ait detaylar Şekil 4.2 ve Şekil 4.3' de yer almaktadır.

Çizelge 4.1. Kromozom yapısı

1. Gen	2. Gen	3. Gen	4. Gen	…….	n. Gen
11 91 50 31 0	31 21 10 32 0	0 32 70 81 91	11 81 40 33 0	…….	I_1 I_2 F O_1 O_2



Şekil 4.2. Lojik kapı detayı



Şekil 4.3. Hafıza elemanı detayı

I_1 ve I_2, iki girişli bir kapıya ait girişleri simgelemektedir. Girişlere ait sayı aralıkları ve taşıdığı anlamlar Çizelge 4.2' de yer almaktadır.

Çizelge 4.1’ de görüldüğü üzere rasgele seçilen fonksiyon bir lojik kapı veya hafıza elemanını simgeliyorsa fonksiyon girişleri, kombinasyonel devre girişleri, kombinasyonel devre girişlerinin tersi, bir önceki lojik kapının çıkışı (O_1), hafıza elemanının çıkışı (Q) veya hafıza elemanın çıkışının tersi (Q’) olabilmektedir.

Çizelge 4.2. Fonksiyon girişleri ve alabilecekleri değer aralıkları

Gen içerikleri	Alabileceği değer aralıkları
1. Giriş (I1)	Kombinasyonel Devre Girişleri : 11....19 Kombinasyonel Girişlerin Değili : 21....29 Lojik Kapı Çıkışları : 31.....79 Hafıza Elemanı Çıkışları : 81....89 Hafıza Elemanı Çıkışlarının Tersi : 91....99
2. Giriş (I2)	Kombinasyonel Devre Girişleri : 11....19 Kombinasyonel Girişlerin Değili : 21....29 Lojik Kapı Çıkışları : 31.....79 Hafıza Elemanı Çıkışları : 81....89 Hafıza Elemanı Çıkışlarının Tersi : 91....99
Fonksiyon (F)	10 (AND) 20 (OR) 30 (EXOR) 40 (NAND) 50 (NOR) 60 (EXNOR) 70 (D tipi Flip-Flop)
1. Çıkış (O1)	Lojik Kapı Çıkışları : 31.....79 Hafıza Elemanı Çıkışları : 81....89
2. Çıkış (O2)	Hafıza Elemanı Çıkışlarının Tersi : 91....99 (F) Lojik Kapı ise : 0

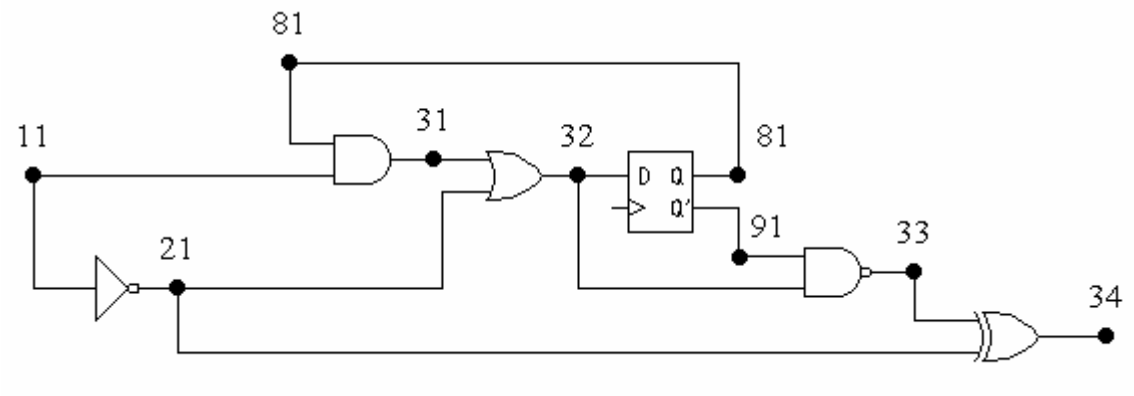
Eğer fonksiyon, bir lojik kapıyı simgeliyorsa O_1 “31..79” arası değer alırken O_2, sıfır değerini alacaktır. Çünkü lojik kapı tek bir çıkışa sahiptir.

Eğer fonksiyon, bir hafıza elemanını simgeliyorsa O_1, “81..89” arası değer alırken O_2, “91..99” arası değer alabilecektir. Çünkü hafıza elemanı Q ve Q’ olmak üzere 2 adet çıkışa sahiptir.

Çizelge 4.3’ de 5 genli bir kromozom yapısı ve bunun ifade ettiği devre yapısı Şekil 4.4’ de yer almaktadır.

Çizelge 4.3. Beş genli kromozom yapısı

1. Gen	2. Gen	3. Gen	4. Gen	5. Gen
11 81 10 31 0	31 21 20 32 0	0 32 70 81 91	91 32 40 33 0	33 21 30 34 0



Şekil 4.4. Kromozom yapısına ait devre şeması

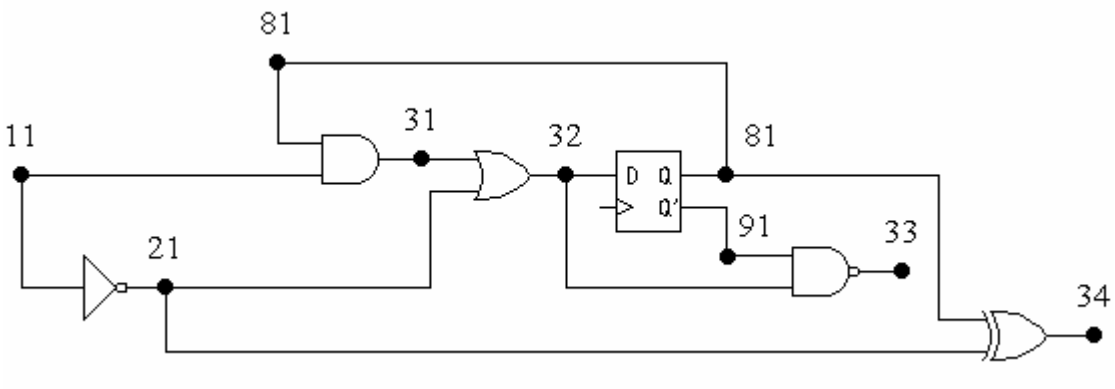
GA'nın hazırlık aşamasında belirlenen kombinasyonel devre girişi adedi, Flip Flop adedi ve gen sayısına göre n adet rasgele kromozom üretilir.

4.2.1.2. Geri yayılım

Bu aşamada kromozom içerisinde yer alan gereksiz genler yok edilerek kromozomların en sade halleri elde edilir. Geri yayılım işlemine alınmamış 5 genli bir kromozomun içeriği Çizelge 4.4' de yer almaktadır. Bu kromozomun ifade ettiği devre Şekil 4.5' de görüldüğü gibidir.

Çizelge 4.4. Beş genli geri yayılması yapılmamış kromozom

1. Gen	2. Gen	3. Gen	4. Gen	5. Gen
11 81 10 31 0	31 21 20 32 0	0 32 70 81 91	91 32 40 33 0	81 21 30 34 0

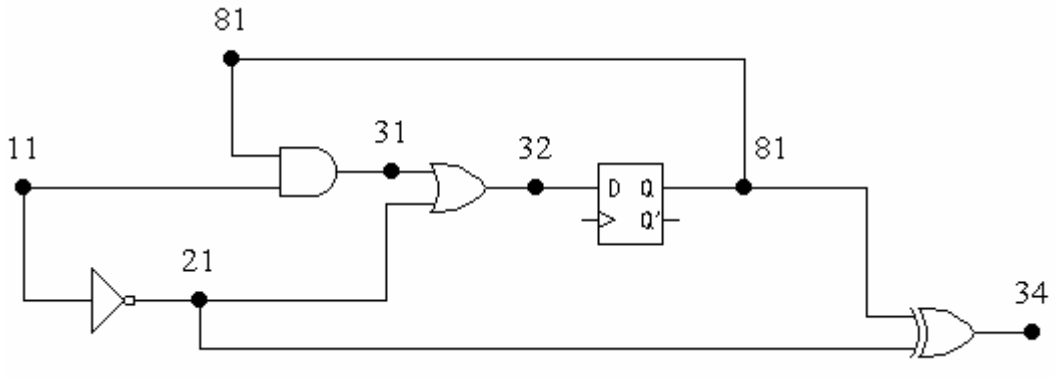


Şekil 4.5. Geri yayılımı yapılmamış kromozom yapısına ait devre şeması

Şekil 4.5' de görüldüğü gibi "4.gen", 34 numaralı çıkışın değerini etkilememektedir. Bu yüzden kromozom yapısında 4.genin yer aldığı kısma "-1" yazılır ve böylece kromozom sadeleştirilmiş olur. Geri yayılım işlemi sonrasındaki kromozomun içeriği Çizelge 4.5' de görülmektedir. Bu kromozomun ifade ettiği devre Şekil 4.6' da görüldüğü gibidir.

Çizelge 4.5. Geri yayılımı yapılmış kromozom yapısı

1. Gen	2. Gen	3. Gen	4. Gen	5. Gen
11 81 10 31 0	31 21 20 32 0	0 32 70 81 91	-1 -1 -1 -1 -1	81 21 30 34 0

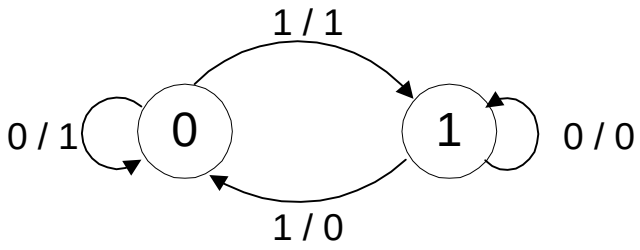


Şekil 4.6. Geri yayılımı yapılmış kromozom yapısına ait devre şeması

4.2.1.3. Hesaplama ve Uygunluk Fonksiyonu

Flip-Flop ve giriş sayısına bağlı olarak "Önceki durum (present state)" ve giriş değerlerinin durum tablosu veya durum diyagramı oluşturulur. Elde edilmek istenen sonraki durumlar ve çıkışlar belirlenir. Bu veriler ile her bir kromozomun oluşturduğu devreden elde edilecek sonraki durum ve çıkış değerleri kıyaslanır. Değerlerin karşılaştırılması esnasında birebir tutan değerlerin varlığı uygunluk değerini arttıracaktır.

Örneğin Şekil 4.7' de görünen durum diyagramına sahip D Flip-Floplu bir ardışık lojik devreyi GA ile tasarlayalım. Öncelikle durum tablosunun oluşturulması gerekmektedir.



Şekil 4.7. Durum diyagramı

Çizelge 4.6. Durum tablosu

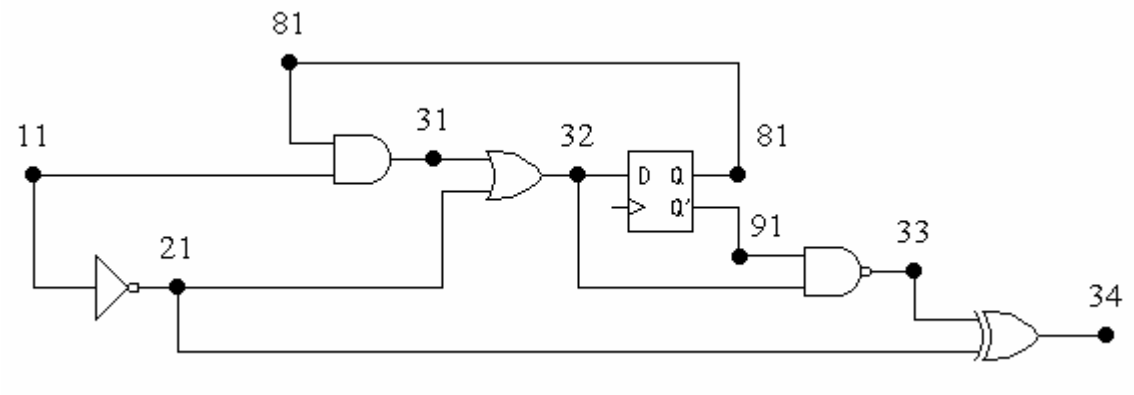
Önceki Durum (Present State)	Giriş (Input)	Sonraki Durum (Next State)	Çıkış (Output)
0	0	0	1
0	1	1	1
1	0	1	0
1	1	0	0

Şekil 4.7' de yer alan durum diyagramı Çizelge 4.6' da görüldüğü gibi durum tablosuna dönüştürülür. Bu işlem sonrasında rasgele n sayıda kromozom üretilir.

Çizelge 4.7. GA tarafından rasgele üretilen kromozom

1. Gen	2. Gen	3. Gen	4. Gen	5. Gen
11 81 10 31 0	31 21 20 32 0	0 32 70 81 91	91 32 40 33 0	33 21 30 34 0

GA tarafından rasgele üretilen kromozomlardan bir tanesinin Çizelge 4.7' de görüldüğü gibi olduğunu düşünelim. Bu kromozomun içerdiği devre Şekil 4.8' de olduğu gibidir. Acaba Çizelge 4.6'da yer alan durum tablosunu sağlayan devre Şekil 4.8' de görünen devre mi?



Şekil 4.8. GA tarafından rasgele üretilen kromozomun ifade ettiği devre

Bunu bilebilmek için öncelikle kromozomun içerdiği çözüm devreye dönüştürülerek ona ait sonuçlar elde edilmelidir. Bunun gerçekleştirilmesi ya bir elektronik devre simülasyonu yada bizzat tasarımcı tarafından sağlanabilir. EWB v5.12 programı ile elde edilen sonuçlar Çizelge 4.8' de yer almaktadır.

Çizelge 4.8. GA tarafından elde edilen çözümün istenen çözüm ile karşılaştırılması

Önceki Durum (81)	Giriş (11)	Elde edilmek istenen Sonraki Durum	31 numaralı çıkış	32 numaralı çıkış	33 numaralı çıkış	Kromozomdan elde edilen Sonraki Durum	Elde edilmek istenen Çıkış	Kromozomdan elde edilen Çıkış (34)
0	0	**0**	0	1	1	**1**	*1*	*0*
0	1	**1**	0	0	1	**0**	*1*	*1*
1	0	**1**	0	1	1	**1**	*0*	*0*
1	1	**0**	1	1	1	**1**	*0*	*1*

Çizelge 4.8' de elde edilmek istenen sonraki durumlar ile kromozomun oluşturduğu devreden elde edilen sonraki durumlar koyu olarak, elde edilmek istenen çıkışlar ile kromozomun oluşturduğu devreden elde edilen çıkışlar da italik olarak gösterilmektedir. Bunlar kendi aralarında karşılaştırılır ve tutan her değer uygunluk

değerini arttıracaktır. Karşılaştırmada tutan değerler altı çizili olarak gösterilmektedir. Tutması gereken 8 değer varken bunların sadece 3 tanesi benzerlik taşıdığından kromozomun aranan çözüm olmadığı açıktır.

Çözümlerin uygunluluklarının hesaplanmasında kullanılan uygunluk fonksiyonu aşağıda görülmektedir.

Uygunluk fonksiyonu = X * (SDDS) + Y * (ÇDS) + Z / (TLKS).....(4.1)

(4.1) deki eşitlikte yer alan SDDS, Sonraki Durum Doğruluk Sayısını; ÇDS, Çıkış Doğruluk Sayısını ve TLKS de Toplam Lojik Kapı Sayısını ifade etmektedir.Ayrıca, fonksiyonda yer alan X, Y, Z katsayıları doğruluk sayılarının ve toplam kapı sayısının uygunluk fonksiyonu içerisindeki ağırlıklarını belirlemektedir.

Uygunluk fonksiyonunun içeriğine bakıldığında en az sayıda kapı kullanarak en fazla sayıda doğruluk elde etme amaçlanmaktadır. Bu durumda GA'nın başarısı uygunluk fonksiyonunun çalışmasına bağlıdır.

4.2.1.4. Seçim

Hesaplama basamağında uygunlukları hesaplanan kromozomlardan bazıları seçim yöntemiyle elenirler. Bölüm 2'de bu durum detaylı olarak ele alınmıştır.

Verilen şartları sağlayan (sonraki durum ve çıkış verileri) en uygun ardışık lojik devre çözümünün bulunmasına dair problemde iki parametreli turnuva metodu kullanıldığını düşünelim. Çözüm topluluğu büyüklüğünün n kromozom olduğu varsayılırsa, rasgele yapılacak n farklı çift seçiminde, uygunluk değeri daha büyük olan çözüm, yeni çözüm topluluğu içerisinde yerini alacaktır.

Turnuva metodu ile yapılan elemelerde zayıf (düşük uygunluk değerine sahip) olan çözüm turnuvayı kaybederken güçlü olan çözüm (daha yüksek uygunluk değerine sahip) yeni çözüm topluluğunda yerini alacaktır.

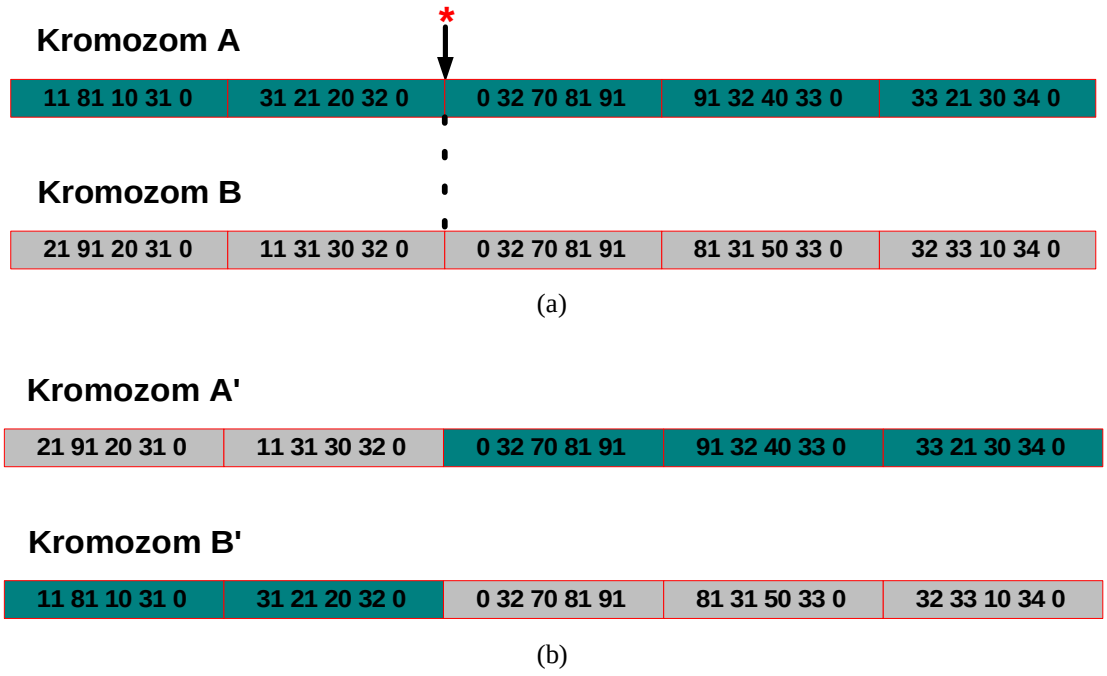
4.2.1.5. Çaprazlama

Çaprazlamada amaç önceden elde edilmiş bilgilerin yeni gruplara aktarılmasını sağlamaktır. Bu aşamada, kromozom genlerinin yerleri değiştirilerek yavru

kromozomlar üretilir ve böylece uygunluk değeri yüksek olan ana kromozomlardan, uygunluk değeri daha yüksek olan yavru kromozomlar elde edilmesi amaçlanır.

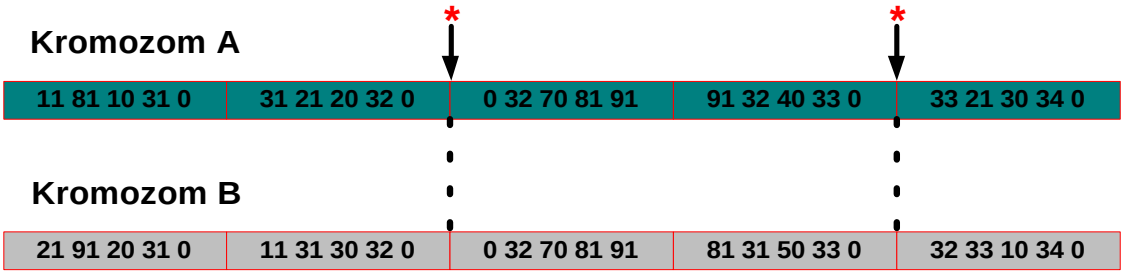
Seçim aşamasında elenmeyen kromozomlar çaprazlama yapmak için seçilirler. Kromozomlar rasgele ikişer ikişer seçilirler ve çaprazlama işlemine tabii tutulurlar.

Verilen şartları sağlayan (sonraki durum ve çıkış verileri) en uygun ardışık lojik devre çözümünün bulunmasına dair problemde tek noktalı çaprazlama işlemi yapıldığını düşünelim. Çaprazlama noktası rasgele belirlenecek olursa iki farklı kromozomdan yeni iki farklı kromozom elde edilecektir. Rasgele seçilen çaprazlama noktası yıldız ile gösterilmiş ok işaretinin bulunduğu noktadır. Tek noktalı çaprazlama işlemi öncesi ve sonrası kromozomlar Şekil 4.9.a ve Şekil 4.9.b' de görülmektedir.

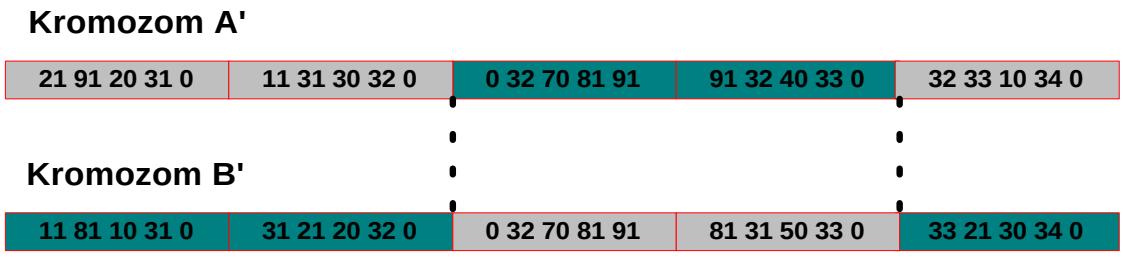


Şekil 4.9.a) Tek noktalı çaprazlama işlemi öncesi b) Tek noktalı çaprazlama işlemi sonrası

Çaprazlama işlemi eğer iki noktadan yapılmak istenirse rasgele seçilecek iki farklı noktadan bölünmeler yıldız ile gösterilmiş ok işaretlerinin bulunduğu noktalardan Şekil 4.10.a ve Şekil 4.10.b' de görüldüğü gibi gerçekleşir.



(a)



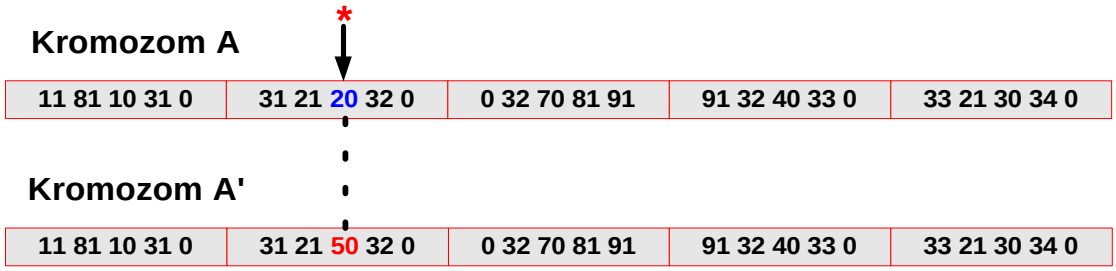
(b)

Şekil 4.10. a) İki noktalı çaprazlama işlemi öncesi b) İki noktalı çaprazlama işlemi sonrası

Çaprazlama noktası sayısını arttırmak mümkün olabilmektedir. Bu durum elde edilecek yavru çözümlerin daha da karılmış bir yapıya sahip olmasına neden olur.

4.2.1.6. Mutasyon

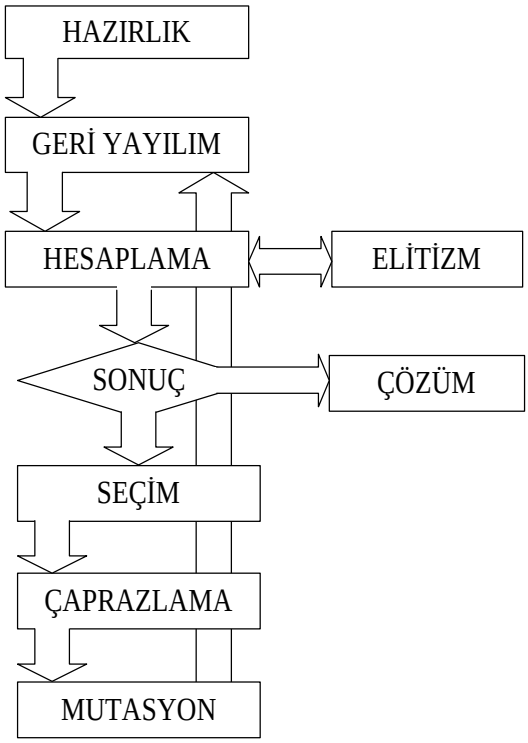
Çaprazlama işlemi sonucunda bireyler birbirine benzemeye başlayabilir. Bu durumdan kurtulabilmesi için Mutasyon uygulanması gerekir. Mutasyon işlemi topluluktaki, problemi çözme yeteneği iyi olmayan ve birbirine benzer bireyleri değiştirerek yeni alt çözümler elde etmek için kullanılır (KERT, 2006). Mutasyon işlemi, rasgele seçilen kromozomda rasgele seçilen bir fonksiyon (F), birinci giriş (I_1) veya 2. giriş (I_2) bilgisinin yerine yine rasgele belirlenecek bir fonksiyon (F), birinci giriş (I_1) veya 2. giriş (I_2) bilgisinin yerini alması şeklinde yapılmıştır. Şekil 4.11' de gösterildiği gibi "Kromozom A" da rasgele seçilen ve ok ile belirtilmiş içerik rasgele belirlenen bir başka değer ile yerleri değiştirilerek Şekil 4.11' de görünen yeni Kromozom A' elde edilmiştir.



Şekil 4.11. Mutasyon öncesi ve mutasyon sonrası

4.2.1.7. Elitizm

Çaprazlama ve mutasyon sonucunda en iyi uyumluluğa sahip birey seçilmeyebilir. Bunu önlemek için, çaprazlama ve mutasyon işlemlerinden sonra oluşan yeni popülasyonun uygunluk değeri düşük herhangi bir kromozomu ile bir önceki popülasyonun en iyi (elit) kromozomu değiştirilir. Buna elitizm adı verilir. GA'ya elitizm eklendiğinde elde edilen akış diyagramı Şekil 4.12' de gösterilmiştir.



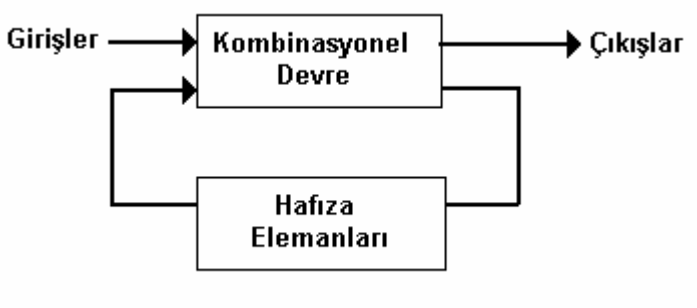
Şekil 4.12. Elitizm eklenmiş GA

Şekil 4.12’ de görünen algoritmada hesaplama ve uygunluk fonksiyonu tarafından elde edilmiş uygunluğu en fazla olan kromozom elitizm operatörü tarafından saklanır. Saklanan kromozom çaprazlama ve mutasyon operatörlerinin çalışmasıyla oluşan kromozomlarla karşılaştırılır. Elitizm operatörü tarafından saklanan kromozom uyumluluğu az olan kromozom ile değiştirilerek elitizm yapılır.

Elitizim, GA’nın bulduğu en iyi çözümün kaybolmasını önleyecektir.

5. ARAŞTIRMA BULGULARI VE TARTIŞMA

Bu çalışmada belli bir sayıda giriş ve tek bir çıkış bulunan, kombinasyonel lojik devre ile hafıza elemanlarını içeren, istenilen durum geçişlerini elde etmeğe çalışan ardışık lojik devreler tasarlanıp optimize edilmiştir. Basit ardışık lojik devre şeması Şekil 5.1' de görüldüğü gibidir.

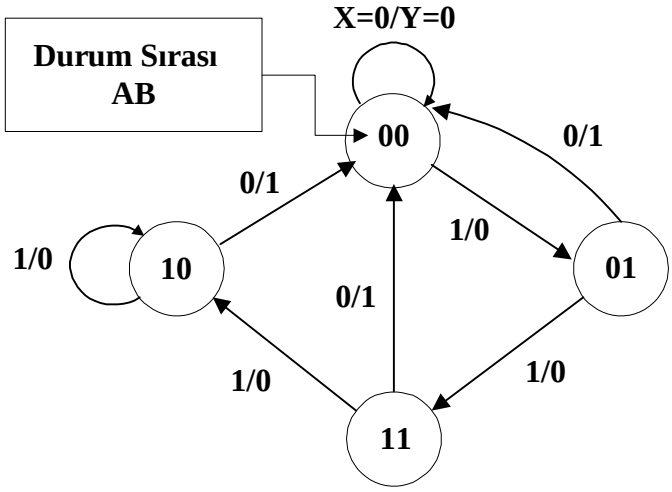


Şekil 5.1. Basit ardışık lojik devre şeması

Optimizasyon gerçekleştirilirken EH yöntemlerinden biri olan GA kullanılmıştır. Öncelikle tasarlanması istenen ardışık lojik devreye ait durum diyagramı veya durum tablosu bilgisi GA'ya aktarılır. Bu bilgiler, rasgelelik esasına göre GA tarafından bulunacak olası devre çözümünün istenen çözüm olup olmadığını kıyaslama açısından kullanılmaktadır.

5.1. GA ile tek girişli ve iki durumlu ardışık lojik devre tasarımı

Şekil 5.2' de durum diyagramı verilen ardışık lojik devreye ait durum tablosu Çizelge 5.1' de görülmektedir.

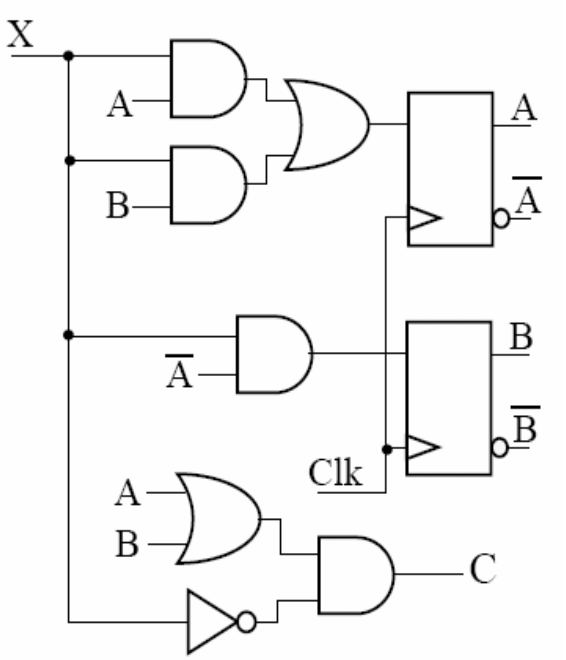


Şekil 5.2. Durum diyagramı

Çizelge 5.1. Durum tablosu

Önceki Durum	Giriş	Sonraki Durum	Çıkış
AB	X	A+B+	C
00	0	00	0
00	1	01	0
01	0	00	1
01	1	11	0
10	0	00	1
10	1	10	0
11	0	00	1
11	1	10	0

Bu veriler doğrultusunda elde edilen devre Şekil 5.3’ de göründüğü gibi olmaktadır (STROUND, 2006).



Şekil 5.3. Ardışık lojik devre (STROUND, 2006)

GA, ilk çalışmada aşağıdaki Çizelge 5.2' de belirtilen parametre değerlerini kullanmıştır.

Çizelge 5.2. İlk çalışmada kullanılan parametre bilgileri

Kromozom Sayısı	10
Gen Sayısı	8
Giriş Sayısı	1
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%30
Jenerasyon Sayısı	5000

Bu veriler doğrultusunda GA, Çizelge 5.3' de belirtilen sonuçları sağlayan devreyi bulmuştur.

Çizelge 5.3. GA'nın ilk çalıştırmada elde ettiği sonuçlar

Önceki durum	Giriş	Elde edilmek istenen sonraki durum	GA tarafından elde edilen sonraki durum	Elde edilmek istenen çıkış	GA tarafından elde edilen çıkış
00	0	00	00	0	0
00	1	01	01	0	0
01	0	00	00	**1**	**0**
01	1	11	11	**0**	**1**
10	0	00	00	**1**	**0**
10	1	10	10	**0**	**1**
11	0	00	00	**1**	**0**
11	1	10	11	**0**	**1**

Bu sonuçlar incelendiğinde "Elde edilmek istenen sonraki durum" değerleri ile "GA tarafından elde edilen sonraki durum" değerleri birebir tutarken ardışık lojik devrenin çıkışında elde edilmek istenen değerlerden sadece ilk iki tanesi tutmaktadır. Tutmayan değerler koyu ve altı çizili olarak görülmektedir. GA'nın bu çalışmasında 8/8 (%100) oranında durum sayısının ve 2/8 (%25) oranında çıkış sayısının tutması dolayısıyla 5000 jenerasyon için elde edilen başarı %62.5 oranında olmaktadır.

Çizelge 5.4. GA'nın ikinci çalışmada kullandığı parametre bilgileri

Kromozom Sayısı	20
Gen Sayısı	8
Giriş Sayısı	1
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%10
Jenerasyon Sayısı	5000

GA ikinci kez Çizelge 5.4' de belirtilen parametreler doğrultusunda çalıştırıldığında ise Çizelge 5.5' de yer alan sonuçları elde edilmiştir.

Çizelge 5.5. GA'nın ikinci çalışmasında elde ettiği sonuçlar

Önceki durum	Giriş	Elde edilmek istenen sonraki durum	GA tarafından elde edilen sonraki durum	Elde edilmek istenen çıkış	GA tarafından elde edilen çıkış
00	0	00	00	0	0
00	1	01	01	0	0
01	0	00	00	1	1
01	1	11	11	0	0
10	0	00	00	1	**0**
10	1	10	10	0	0
11	0	00	00	1	1
11	1	10	11	0	0

Bu sonuçlar incelendiğinde "Elde edilmek istenen sonraki durum" değerleri ile "GA tarafından elde edilen sonraki durum" değerleri birebir tutarken ardışık lojik devrenin çıkışında elde edilmek istenen değerlerden sadece bir tanesi tutmaktadır. Tutmayan değerler koyu ve altı çizili olarak görülmektedir. GA'nın bu çalışmasında 8/8 (%100) oranında durum sayısının ve 7/8 (%87.5) oranında çıkış sayısının tutması dolayısıyla 5000 jenerasyon için elde edilen başarı %93.75 oranına yükselmiştir.

Çizelge 5.6. GA'nın son çalışmada kullandığı parametre bilgileri

Kromozom Sayısı	20
Gen Sayısı	8
Giriş Sayısı	1
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%10
Jenerasyon Sayısı	5000

GA, son kez Çizelge 5.6' da belirtilen parametreler doğrultusunda çalıştırıldığında ise Çizelge 5.7' de yer alan sonuçları elde edilmiştir.

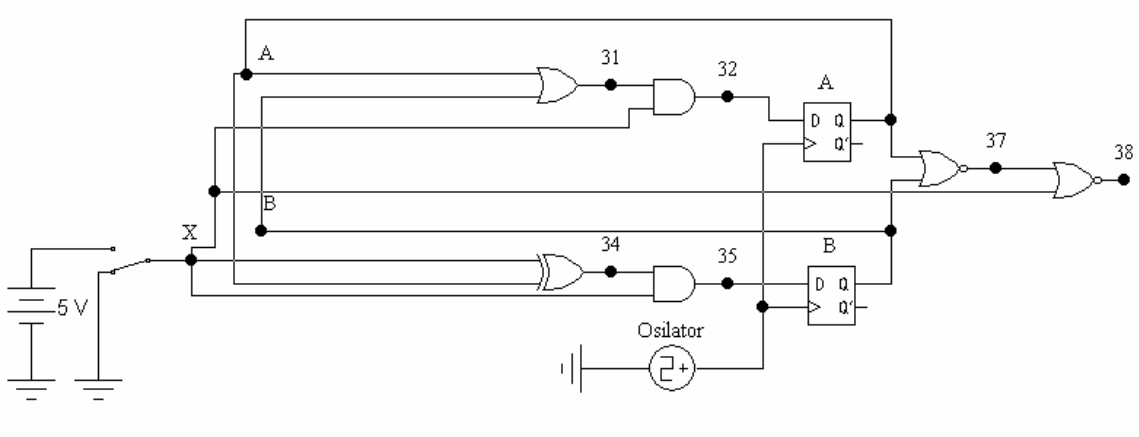
Çizelge 5.7. GA'nın son çalışmasında elde ettiği sonuçlar

Önceki durum	Giriş	Elde edilmek istenen sonraki durum	GA tarafından elde edilen sonraki durum	Elde edilmek istenen çıkış	GA tarafından elde edilen çıkış
00	0	00	00	0	0
00	1	01	01	0	0
01	0	00	00	1	1
01	1	11	11	0	0
10	0	00	00	1	1
10	1	10	10	0	0
11	0	00	00	1	1
11	1	10	11	0	0

Bu sonuçlar incelendiğinde "Elde edilmek istenen sonraki durum" değerleri ile "GA tarafından elde edilen sonraki durum" değerleri ve ardışık lojik devrenin çıkışında elde edilmek istenen değerler birebir tutmaktadır. GA'nın bu çalışmasında 8/8 (%100) oranında durum sayısının ve 8/8 (%100) oranında çıkış sayısının tutması dolayısıyla 5000 jenerasyon için elde edilen başarı %100 oranına yükselmiştir. Bu sonuçlar doğrultusunda elde edilen kromozoma ait genler Çizelge 5.8' de ve devre Şekil 5.4' de göründüğü gibidir.

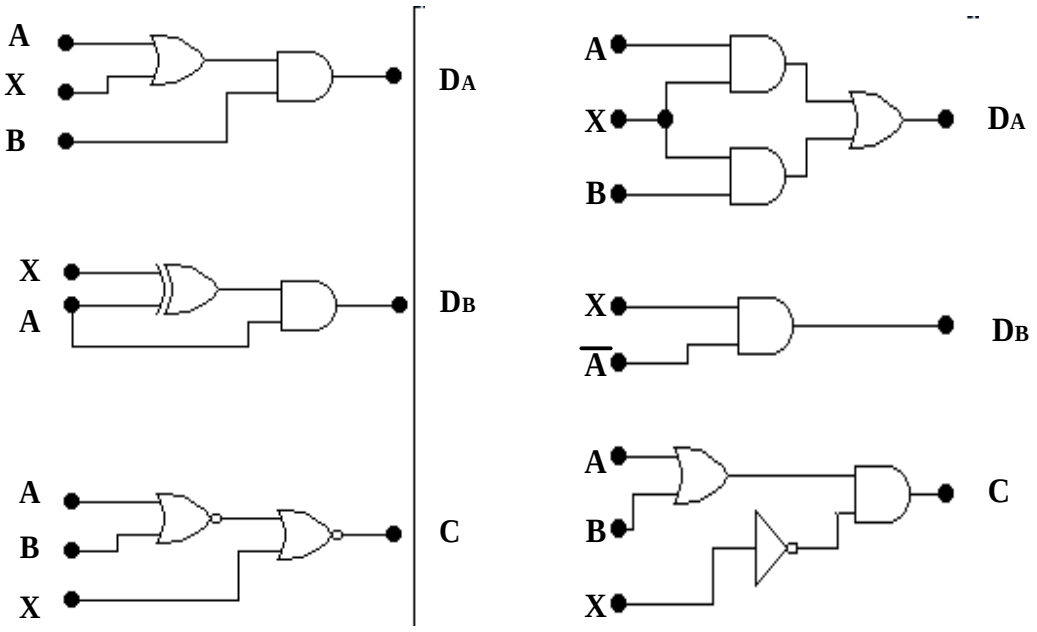
Çizelge 5.8. GA'nın son çalışmasında elde ettiği kromozom

1. Gen	2. Gen	3. Gen	4. Gen
81 82 20 31 0	31 11 10 32 0	0 32 70 81 91	11 81 30 34 0
5. Gen	6. Gen	7. Gen	8. Gen
34 11 10 35 0	0 35 70 82 92	82 81 50 37 0	37 11 50 38 0



Şekil 5.4. GA tarafından üretilen devrenin EWB ortamında tam gösterimi

Sonuç olarak GA ile klasik tasarımın kıyaslaması Şekil 5.5' de ve Çizelge 5.9' da görülmektedir.



(a) (b)

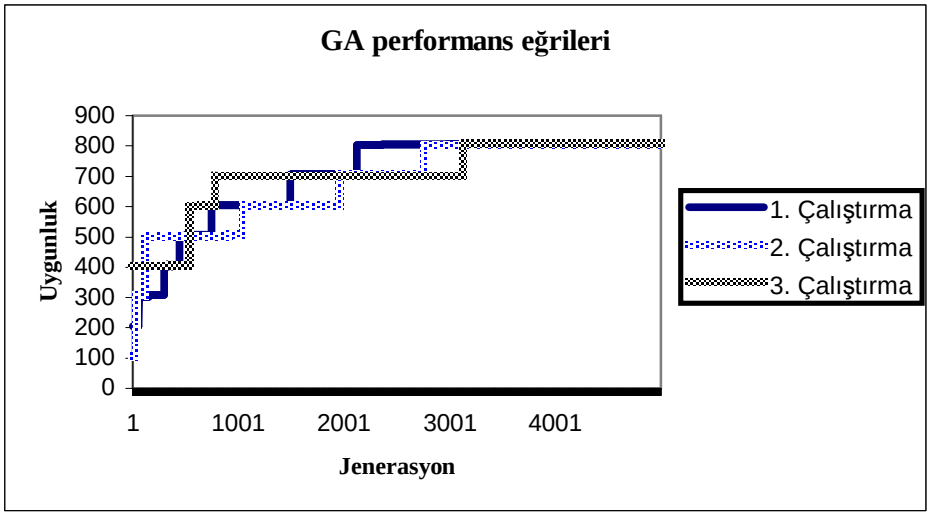
Şekil 5.5. a) GA tasarımı b) Klasik tasarım

Çizelge 5.9. GA ile klasik tasarımın kıyaslanması

	Klasik tasarım	GA ile tasarlanan
FF sayısı	2	2
Lojik kapı sayısı	7	6
Lojik kapı çeşidi	3	4

Tek girişli ve iki durumlu D tipi FF kullanılarak yapılan ardışık lojik devre tasarımına ilişkin elde edilen tasarım sonuçları doğrultusunda, GA ile tasarlanan devrenin klasik tasarımdan 1 kapı daha az kullanılarak gerçekleşmiştir.

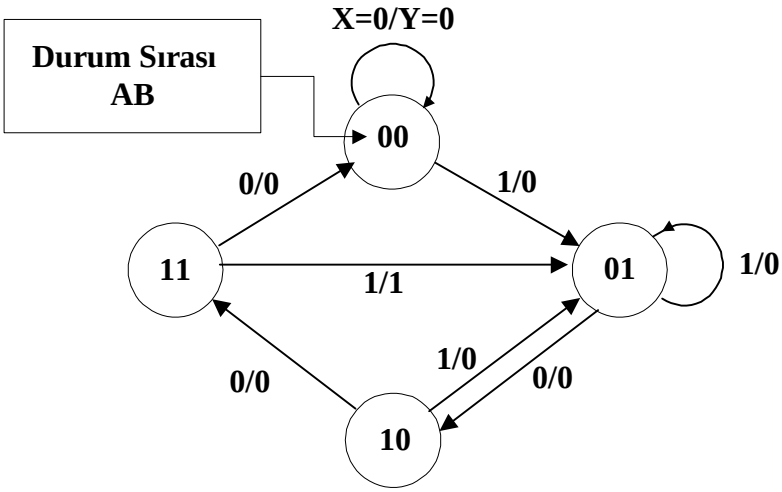
(4.1) numaralı uygunluk fonksiyonundaki X değeri, 100; Y değeri, 1 ve Z değeri de 1 olarak kabul edilmiştir.Elde edilen bu sonuçlara bağlı olarak ortaya çıkan GA'nın 3 farklı çalışmasına ait performans grafiği Şekil 5.6' da görüldüğü gibidir.



Şekil 5.6. GA performans grafiği

Tek girişli ve iki durumlu ardışık lojik devre tasarımına ilişkin ikinci bir örneğe ait ayrıntılar aşağıda verilmektedir.

Şekil 5.7' de durum diyagramı verilen ardışık lojik devreye ait durum tablosu (ANONYMUS, 2005) Çizelge 5.10' da görülmektedir.



Şekil 5.7. Durum diyagramı

Çizelge 5.10. Durum tablosu

Önceki Durum	Giriş	Sonraki Durum	Çıkış
AB	X	A+B+	C
00	0	00	0
00	1	01	0
01	0	10	0
01	1	01	0
10	0	11	0
10	1	01	0
11	0	00	0
11	1	01	1

Bu veriler doğrultusunda Çizelge 5.11’ deki parametreler uygulanarak Çizelge 5.12’ deki sonuçlar elde edilmektedir.

Çizelge 5.11. İlk çalışmada kullanılan parametre bilgileri

Kromozom Sayısı	30
Gen Sayısı	21
Giriş Sayısı	1
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%10
Jenerasyon Sayısı	5000

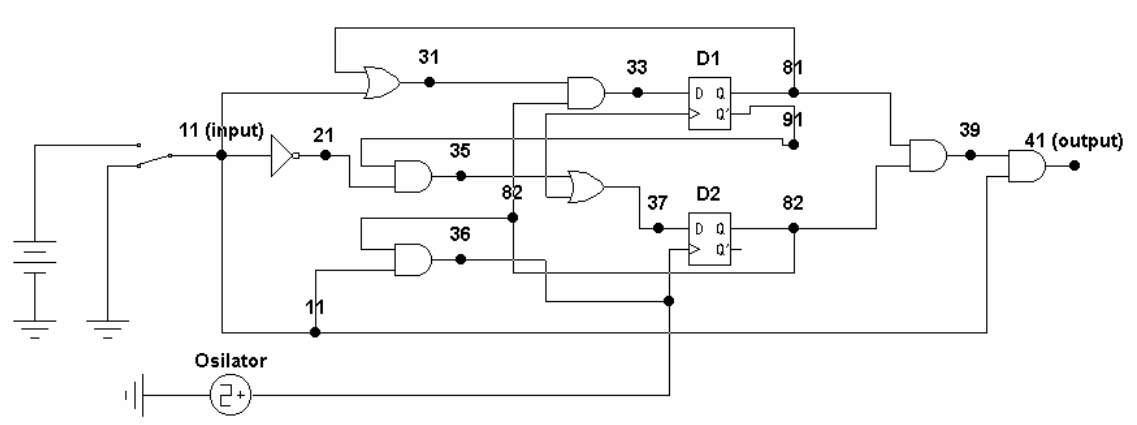
Çizelge 5.12’ de yer alan sonuçlar incelendiğinde “Elde edilmek istenen sonraki durum” değerleri ile “GA tarafından elde edilen sonraki durum” değerleri ve ardışık lojik devrenin çıkışında elde edilmek istenen değerler birebir tutmaktadır. GA’nın bu çalışmasında 8/8 (%100) oranında durum sayısının ve 8/8 (%100) oranında çıkış sayısının tutması dolayısıyla 5000 jenerasyon için elde edilen başarı %100 oranına yükselmiştir. Bu sonuçlar doğrultusunda elde edilen kromozoma ait genler Çizelge 5.13’ de ve devre Şekil 5.8’ de görüldüğü gibidir.

Çizelge 5.12. GA’nın ilk çalışmasında elde ettiği sonuçlar

Önceki Durum	Giriş	Elde edilmek istenen Sonraki Durum	GA tarafından elde edilen Sonraki Durum	Elde edilmek istenen Çıkış	GA tarafından elde edilen Çıkış
00	0	00	00	0	0
00	1	01	01	0	0
01	0	10	10	0	0
01	1	01	01	0	0
10	0	11	11	0	0
10	1	01	01	0	0
11	0	00	00	0	0
11	1	01	01	1	1

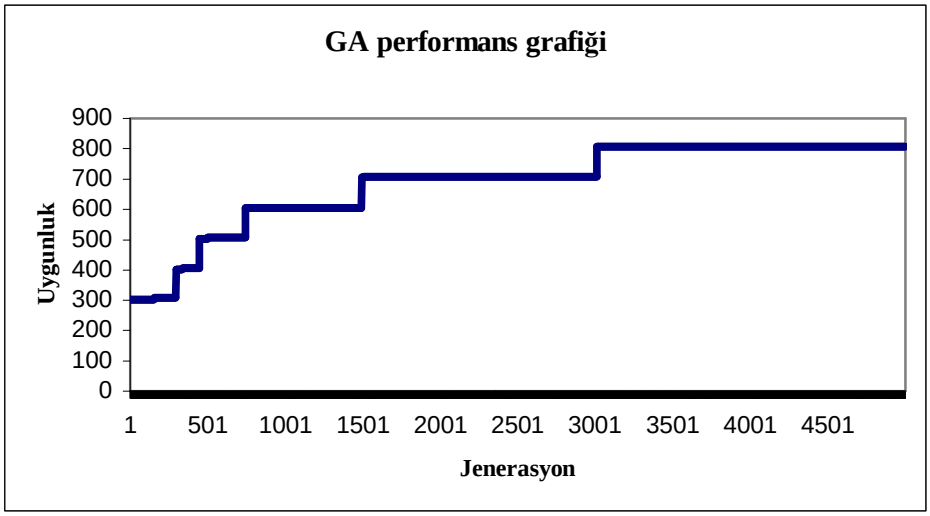
Çizelge 5.13. GA’nın elde ettiği kromozom

1. Gen	2. Gen	3. Gen	4. Gen	5. Gen
81 11 20 31 0	31 82 10 33 0	0 33 70 81 91	21 91 10 35 0	82 11 10 36 0
6. Gen	7. Gen	8. Gen	9. Gen	
34 35 20 37 0	0 37 70 82 92	81 82 10 39 0	11 39 10 41 0	



Şekil 5.8. GA tarafından üretilen devrenin EWB ortamında tam gösterimi

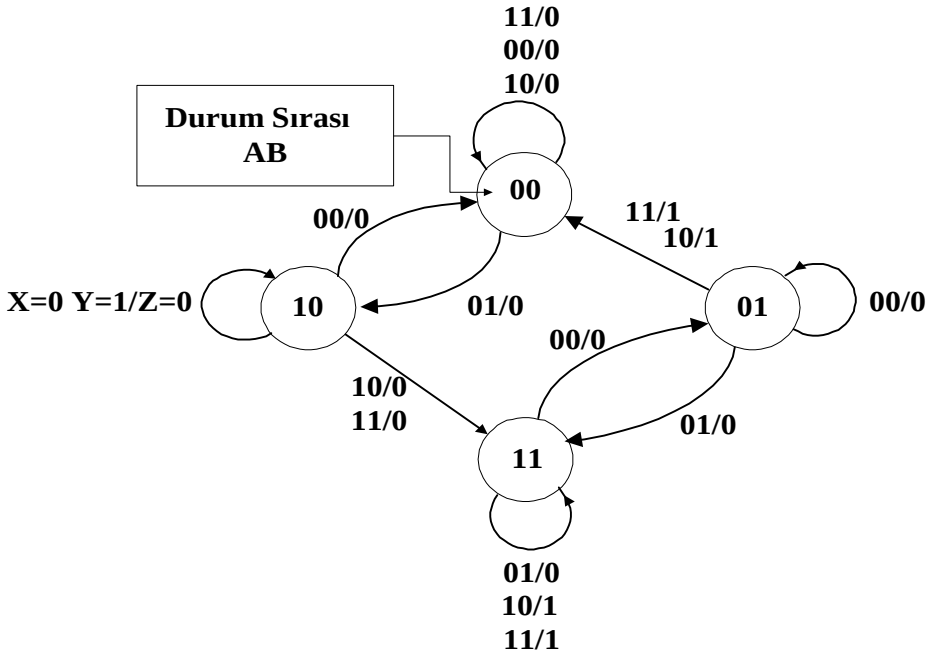
(4.1) numaralı uygunluk fonksiyonundaki X değeri, 100; Y değeri, 1 ve Z değeri de 1 olarak kabul edilmiştir.Elde edilen bu sonuçlara bağlı olarak ortaya çıkan GA'nın çalışmasına ait performans grafiği Şekil 5.9' da görüldüğü gibidir.



Şekil 5.9. GA performans grafiği

5.2. GA ile iki girişli ve iki durumlu ardışık lojik devre tasarımı

Şekil 5.10' da durum diyagramı verilen ardışık lojik devreye ait durum tablosu Çizelge 5.14' de görülmektedir.

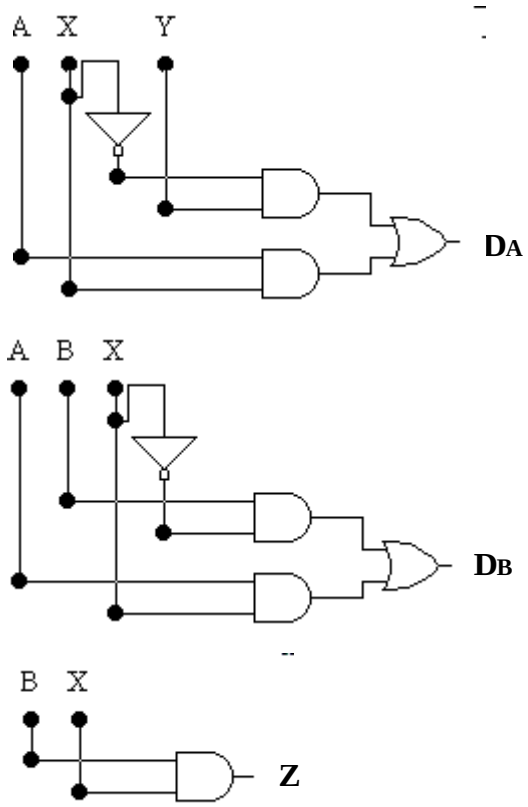


Şekil 5.10. Durum diyagramı

Çizelge 5.14. Durum tablosu

Önceki Durum	Girişler	Sonraki Durum	Çıkış
AB	XY	A+B+	Z
00	00	00	0
00	01	10	0
00	10	00	0
00	11	00	0
01	00	01	0
01	01	11	0
01	10	00	1
01	11	00	1
10	00	00	0
10	01	10	0
10	10	11	0
10	11	11	0
11	00	01	0
11	01	11	0
11	10	11	1
11	11	11	1

Bu veriler doğrultusunda EWB tarafından tasarlanan devre Şekil 5.11’ de göründüğü gibi olmaktadır**.**



Şekil 5.11. EWB tasarımı lojik devre

GA ilk çalışmada Çizelge 5.15' de belirtilen parametre değerleri kullanmıştır.

Çizelge 5.15. İlk çalışmada kullanılan parametre bilgileri

Kromozom Sayısı	50
Gen Sayısı	20
Giriş Sayısı	2
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%10
Jenerasyon Sayısı	10000

Bu veriler doğrultusunda GA, Çizelge 5.16' da belirtilen sonuçları sağlayan devreyi bulmuştur.

Çizelge 5.16. GA'nın elde ettiği sonuçlar

Önceki durum	Giriş	Elde edilmek istenen sonraki durum	GA tarafından elde edilen sonraki durum	Elde edilmek istenen çıkış	GA tarafından elde edilen çıkış
00	00	00	00	0	0
00	01	10	10	0	0
00	10	00	00	0	0
00	11	00	00	0	0
01	00	01	01	0	0
01	01	11	11	0	0
01	10	00	00	1	1
01	11	00	00	1	1
10	00	00	00	0	0
10	01	10	10	0	0
10	10	11	11	0	0
10	11	11	11	0	0
11	00	01	01	0	0
11	01	11	11	0	0
11	10	11	11	1	1
11	11	11	11	1	1

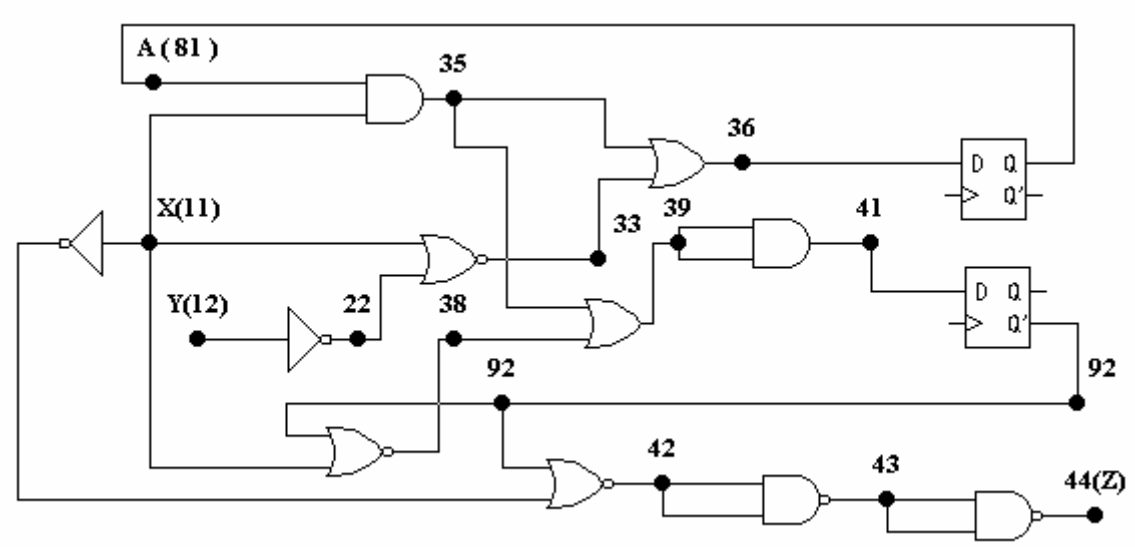
Bu sonuçlar incelendiğinde "Elde edilmek istenen sonraki durum" değerleri ile "GA tarafından elde edilen sonraki durum" değerleri birebir tutarken ardışık lojik devrenin çıkışında elde edilmek istenen değerlerden 16 tanesi tutmaktadır. GA'nın bu çalışmasında 16/16 (%100) oranında durum sayısının ve 16/16 (%100) oranında çıkış sayısının tutması dolayısıyla 10000 jenerasyon için elde edilen başarı %100 oranında olmaktadır.

Çizelge 5.17, GA tarafından elde edilen kromozomu göstermektedir.

Çizelge 5.17. GA'nın elde ettiği kromozom

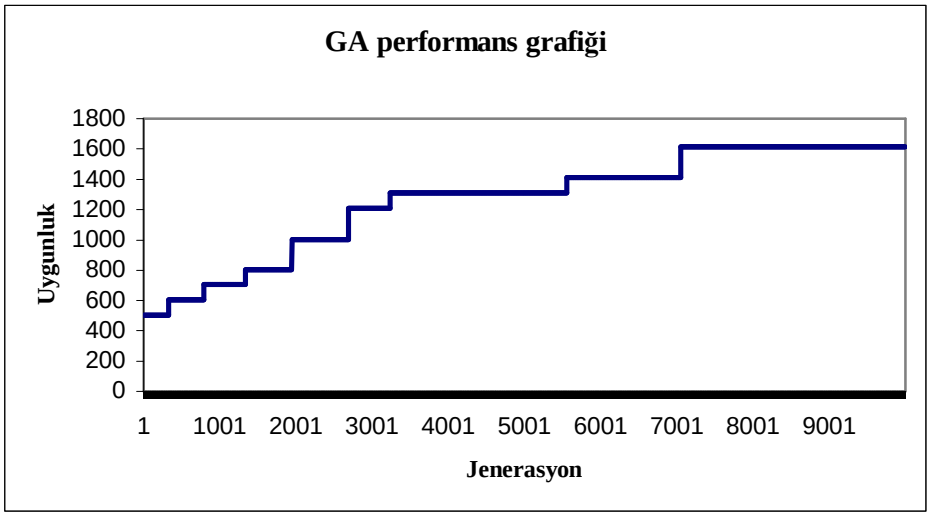
1. Gen	2. Gen	3. Gen	4. Gen
11 22 50 33 0	81 11 10 35 0	35 33 20 36 0	0 36 70 81 91
5. Gen	6. Gen	7. Gen	8. Gen
92 11 50 38 0	35 38 20 39 0	39 39 10 41 0	0 41 70 82 92
9. Gen	10. Gen	11. Gen	
92 21 50 42 0	42 42 40 43 0	43 43 40 44 0	

Şekil 5.12, Çizelge 5.17' de elde edilen kromozomun EWB ortamındaki lojik devre eşdeğerini göstermektedir.



Şekil 5.12. GA tarafından üretilen devrenin EWB ortamında tam gösterimi

(4.1) numaralı uygunluk fonksiyonundaki X değeri, 100; Y değeri, 1 ve Z değeri de 1 olarak kabul edilmiştir.Elde edilen bu sonuçlara bağlı olarak ortaya çıkan GA'nın çalışmasına ait performans grafiği Şekil 5.13' de görüldüğü gibidir.

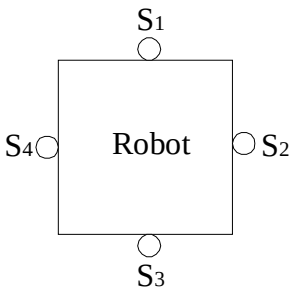


Şekil 5.13. GA performans grafiği

5.3. GA ile dört girişli ve iki durumlu ardışık lojik devre tasarımı

Labirent içerisinde duvara çarpmadan ilerlemeye çalışan bir robot düşünülsün. Bu robotun hareketi sırasında karşısına çıkacak engellere çarpmadan ilerleyebilmesi için

4 farklı sensör kullandığı kabul edilsin. Üzerine sensörleri yerleştirilmiş iki tekerlekli robotun üstten görünüşünün Şekil 5.14' de göründüğü gibi olduğu kabul edildiğinde bu robotun hareketi için gerekli olası durum tablosu Çizelge 5.18' deki gibi olacaktır.



Şekil 5.14. Robotun sensörlerinin konumu

Çizelge 5.18. Dört girişli ve iki durumlu ardışık lojik devrenin durum tablosu

Önceki Durum	Girişler	Sonraki Durum	Çıkış
AB	$S_1S_2S_3S_4$	A+B+	Z
00	0000	11	1
00	0001	01	1
00	0010	11	1
00	0011	01	1
00	0100	11	1
00	0101	01	1
00	0110	11	1
00	0111	01	1
00	1000	11	1
00	1001	10	1
00	1010	11	1
00	1011	11	0
00	1100	11	1
00	1101	10	1
00	1110	11	1
00	1111	00	1
01	0000	11	1
01	0001	10	1
01	0010	11	1
01	0011	01	0
01	0100	11	1

Çizelge 5.18. (Devam) Dört girişli ve iki durumlu ardışık lojik devrenin durum tablosu

AB	$S_1S_2S_3S_4$	A+B+	Z
01	0101	01	1
01	0110	01	1
01	0111	01	1
01	1000	01	1
01	1001	11	1
01	1010	11	1
01	1011	11	0
01	1100	11	1
01	1101	10	1
01	1110	11	1
01	1111	00	1
10	0000	11	1
10	0001	01	1
10	0010	11	1
10	0011	01	1
10	0100	11	1
10	0101	10	1
10	0110	11	1
10	0111	01	1
10	1000	10	1
10	1001	10	1
10	1010	11	1
10	1011	01	1
10	1100	11	1
10	1101	10	1
10	1110	11	1
10	1111	00	1
11	0000	11	1
11	0001	01	1
11	0010	11	1
11	0011	01	1
11	0100	11	1
11	0101	01	1
11	0110	11	1
11	0111	11	1
11	1000	11	1
11	1001	10	1
11	1010	11	1
11	1011	11	0
11	1100	11	1
11	1101	10	1
11	1110	11	1
11	1111	00	0

Çizelge 5.18' deki durumları temsil eden A ve B harfleri, sırasıyla robotun sol ve sağ tekerlerini kontrol eden D tipi flip-floplardır. A değeri *lojik 1* iken teker hareket edecek, *lojik 0* iken duracaktır. B değeri için de aynı durum geçerlidir. S_1, S_2, S_3 ve S_4 ise ana yönlere bakan ve engel olup olmadığını algılamaya yarayan sensörlerdir. Sensörlerden birinin *lojik 1* olması engelin varlığını, *lojik 0* olması ise engelin olmadığı anlamına gelmektedir. Son olarak Z çıkışı ise tekerlerin dönüş yönünü kontrol eden unsurdur. Z çıkışı tekerin hareket yönünü belirlemekte olup *lojik 1* olursa teker ileri yönde hareket edecek, *lojik 0* olduğunda ise geri yönde hareket edecektir.

Çizelge 5.18' deki durum tablosu GA ile elde edilmek istendiğinde Çizelge 5.19' da belirtilen parametreler için Çizelge 5.20' de yer alan sonuçlar ortaya çıkmıştır.

Çizelge 5.19. GA için kullanılan parametre bilgileri

Kromozom Sayısı	100
Gen Sayısı	46
Giriş Sayısı	4
Durum Sayısı	2
Kullanılan FF tipi	D
Çaprazlama Oranı	%50
Mutasyon Oranı	%10
Jenerasyon Sayısı	20000

Çizelge 5.20. GA tarafından elde edilen sonuçlar

Önceki durum	Giriş	Elde edilmek istenen sonraki durum	Elde edilmek istenen çıkış	GA tarafından elde edilen sonraki durum	GA tarafından elde edilen çıkış
AB	$S_1S_2S_3S_4$	A+B+	Z	A+B+	Z
00	0000	11	1	11	1
00	0001	01	1	01	1
00	0010	11	1	11	1
00	0011	01	1	01	1
00	0100	11	1	11	1
00	0101	01	1	**1**1	1
00	0110	11	1	11	1
00	0111	01	1	**1**1	1
00	1000	11	1	11	1
00	1001	10	1	10	1

Çizelge 5.20. (Devam) GA tarafından elde edilen sonuçlar

AB	$S_1S_2S_3S_4$	A+B+	Z	A+B+	Z
00	1010	11	1	11	1
00	1011	11	0	11	0
00	1100	11	1	11	1
00	1101	10	1	10	1
00	1110	11	1	11	1
00	1111	00	1	**1**0	1
01	0000	11	1	11	1
01	0001	10	1	10	1
01	0010	11	1	11	1
01	0011	01	0	01	0
01	0100	11	1	11	1
01	0101	01	1	01	1
01	0110	01	1	01	1
01	0111	01	1	01	1
01	1000	01	1	01	1
01	1001	11	1	11	1
01	1010	11	1	11	1
01	1011	11	0	11	0
01	1100	11	1	11	1
01	1101	10	1	10	1
01	1110	11	1	11	1
01	1111	00	1	**1**0	1
10	0000	11	1	11	1
10	0001	01	1	01	1
10	0010	11	1	11	1
10	0011	01	1	01	1
10	0100	11	1	11	1
10	0101	10	1	10	1
10	0110	11	1	11	1
10	0111	01	1	01	1
10	1000	10	1	1**1**	1
10	1001	10	1	1**1**	1
10	1010	11	1	11	1
10	1011	01	1	**1**1	1
10	1100	11	1	11	1
10	1101	10	1	10	1
10	1110	11	1	11	1
10	1111	00	1	**1**0	1
11	0000	11	1	11	1
11	0001	01	1	01	1
11	0010	11	1	11	1
11	0011	01	1	01	1
11	0100	11	1	11	1
11	0101	01	1	**1**1	1

Çizelge 5.20. (Devam) GA tarafından elde edilen sonuçlar

AB	$S_1S_2S_3S_4$	A+B+	Z	A+B+	Z
11	0110	11	1	11	1
11	0111	11	1	11	1
11	1000	11	1	11	1
11	1011	11	0	11	0
11	1100	11	1	11	1
11	1101	10	1	10	1
11	1110	11	1	11	1
11	1111	00	**0**	**1**0	**1**

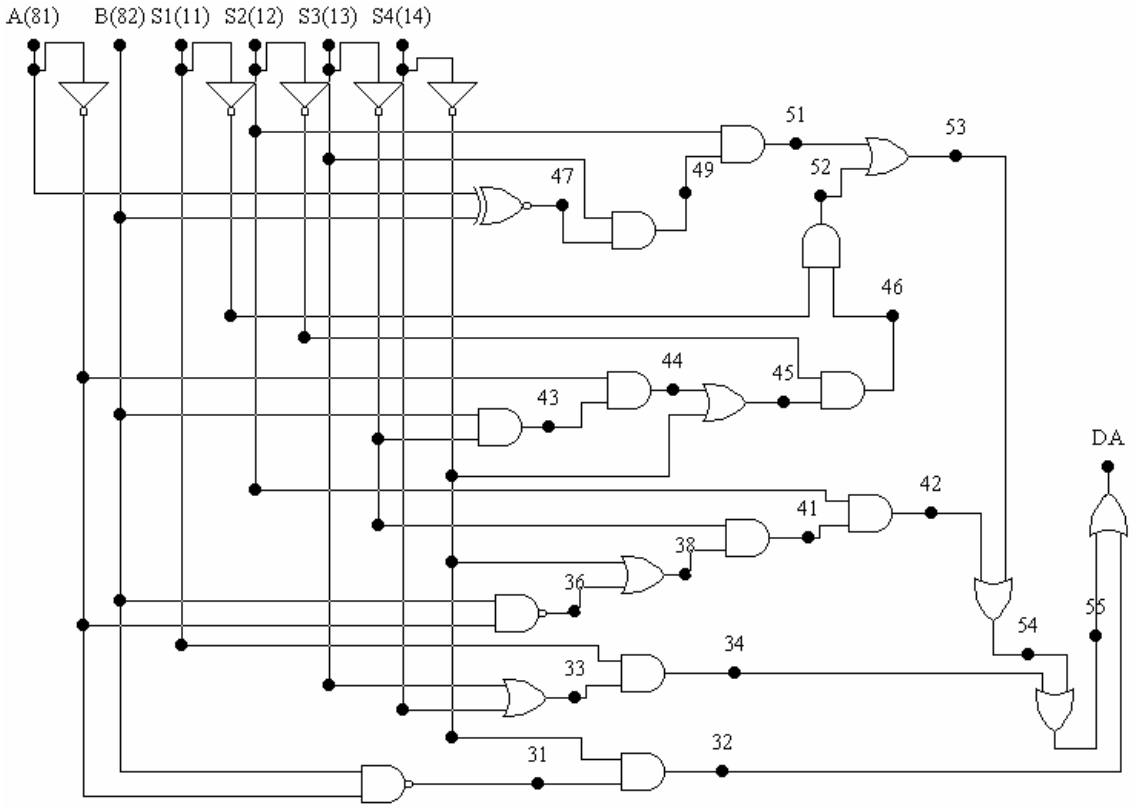
Tasarım üç farklı aşamadan oluşmaktadır. Birincisi, D_A flip-flopuna ait giriş kombinasyonel devresinin elde edilmesi; ikincisi, D_B flip-flopuna ait giriş kombinasyonel devresinin elde edilmesi ve son olarak çıkışa ait kombinasyonel devrenin elde edilmesi şeklindedir. Yani kısmi kombinasyonel devre çözümleri yapılmaktadır.

Çizelge 5.20' de yer alan bu veriler, elde edilemek istenen ve gerçekte elde edilen verileri içermektedir. Bu verilerden koyu ve altı çizili olanlar gerçekte elde edilmek istenip elde edilemeyen verileri ifade etmektedirler.

D_A flip-flopuna ait giriş kombinasyonel devresini ifade eden kromozom yapısı Çizelge 5.21' de yer aldığı gibidir. Bu kromozomdan elde edilen kombinasyonel lojik devrenin şekli Şekil 5.15' de olduğu gibidir.

Çizelge 5.21. GA tarafından üretilen D_A flip-flopu girişine ait kromozom yapısı

1. Gen	2. Gen	3. Gen	4. Gen
91 82 40 31 0	24 31 10 32 0	13 14 20 33 0	11 33 10 34 0
5. Gen	6. Gen	7. Gen	8. Gen
91 82 40 36 0	24 36 20 38 0	23 38 10 41 0	12 41 10 42 0
9. Gen	10. Gen	11. Gen	12. Gen
23 82 10 43 0	91 43 10 440	24 44 20 45 0	22 45 10 46 0
13. Gen	14. Gen	15. Gen	16. Gen
81 82 60 47 0	13 47 10 49 0	12 49 10 51 0	46 21 10 52 0
17. Gen	18. Gen	19. Gen	20. Gen
51 52 20 53 0	42 53 20 54 0	34 54 20 55 0	32 55 20 56 0

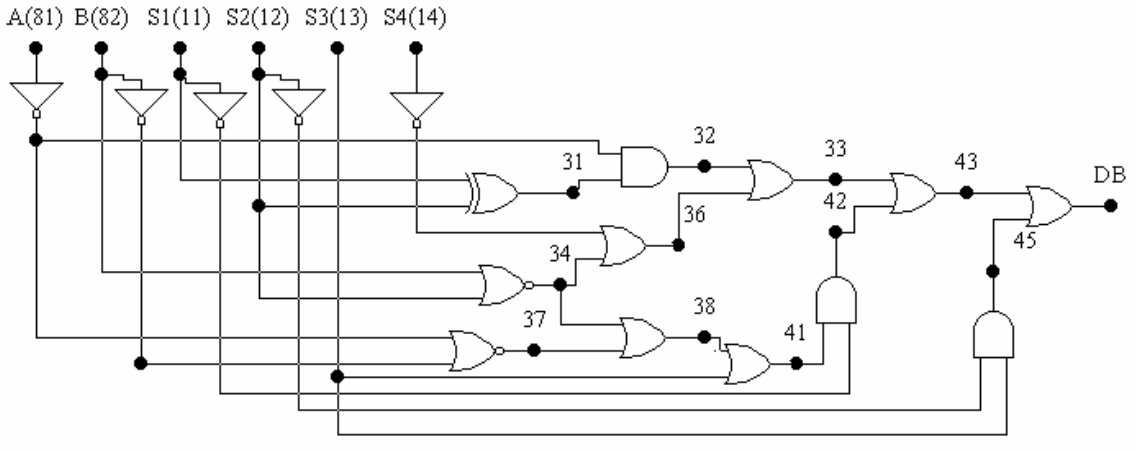


Şekil 5.15. GA tarafından üretilen A flip-flopu girişine ait kombinasyonel lojik devre

D_B flip-flopuna ait giriş kombinasyonel devresini ifade eden kromozom yapısı Çizelge 5.22' de olduğu gibidir. Bu kromozomdan elde edilen kombinasyonel lojik devrenin şekli Şekil 5.16' da olduğu gibidir.

Çizelge 5.22. GA tarafından üretilen D_B flip-flopu girişine ait kromozom yapısı

1. Gen	2. Gen	3. Gen	4. Gen
11 12 30 31 0	91 31 10 32 0	82 12 50 33 0	24 33 20 34 0
5. Gen	6. Gen	7. Gen	8. Gen
32 34 20 35 0	91 92 50 37 0	33 37 20 38 0	13 38 20 41 0
9. Gen	10. Gen	11. Gen	12. Gen
21 41 10 42 0	35 42 20 43 0	13 22 10 45 0	43 45 20 56 0

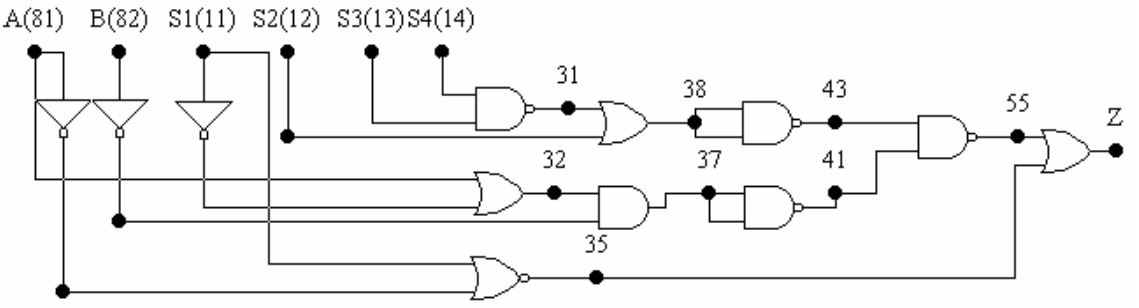


Şekil 5.16. GA tarafından üretilen B flip-flopu girişine ait kombinasyonel lojik devre

Çıkışa ait giriş kombinasyonel devresini ifade eden kromozom yapısı Çizelge 5.23' deki gibidir. Bu kromozomdan elde edilen kombinasyonel lojik devrenin şekli Şekil 5.17' de olduğu gibidir.

Çizelge 5.23. GA tarafından üretilen çıkışa ait kromozom yapısı

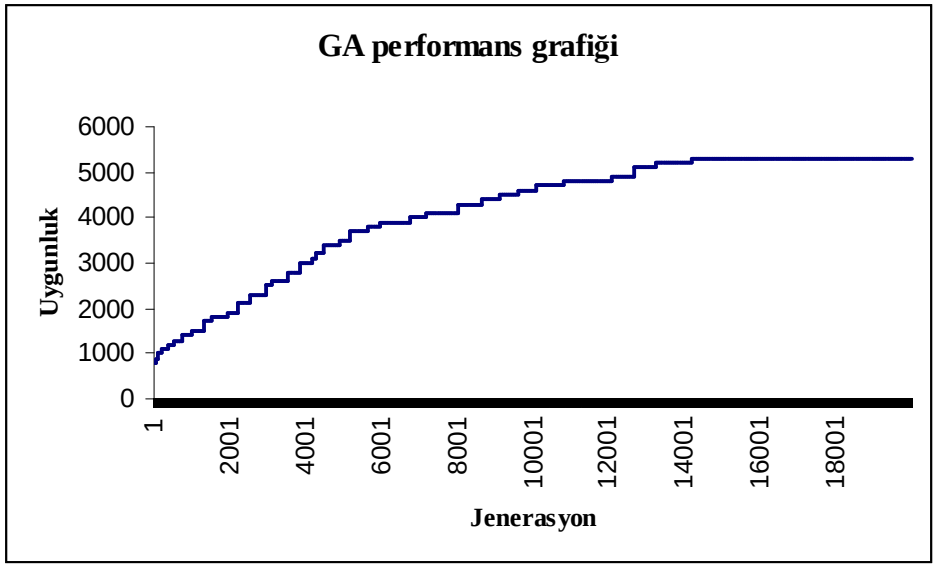
1. Gen	2. Gen	3. Gen	4. Gen
13 14 40 31 0	21 81 20 32 0	11 91 50 35 0	12 31 20 38 0
5. Gen	6. Gen	7. Gen	8. Gen
32 35 40 41 0	38 38 40 43 0	41 43 40 55 0	35 55 20 56 0



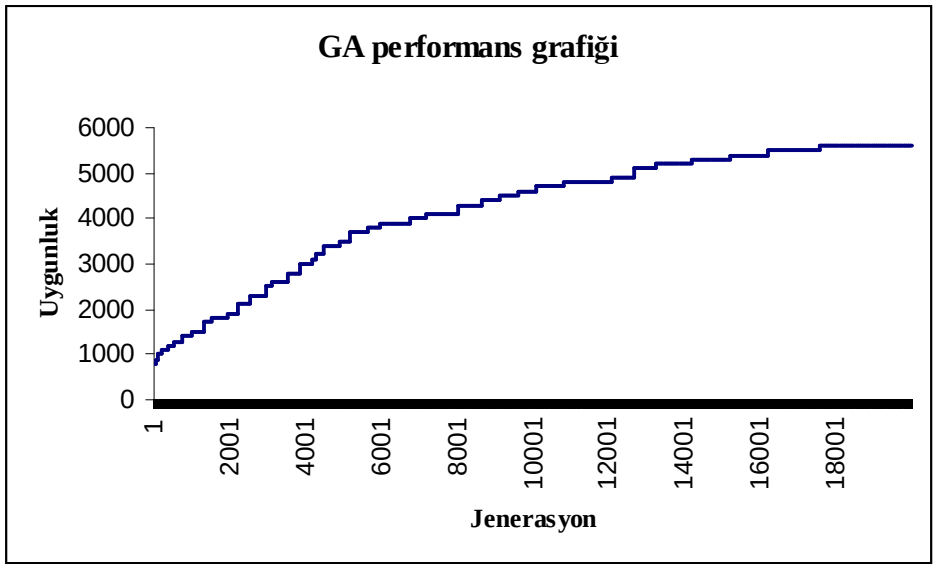
Şekil 5.17. GA tarafından üretilen çıkışa ait kombinasyonel lojik devre

Bu sonuçlar incelendiğinde D_A flip-flopu için elde edilen kombinasyonel lojik devre elde edilmek istenen sonraki durum tablosundan 8 adet durumu karşılamamaktadır. Bu durumda GA tarafından elde edilen bu devrenin başarısı 56/64 yani %87,5 olmaktadır. Bu sonuç elde edilirken 200 jenerasyon boyunca uygunluk

değerinin aynı kalması durumunda mutasyon oranını %1 arttıran ***adaptif mutasyon*** ve 1000 jenerasyon boyunca uygunluk değerinin aynı kalması durumunda kromozomları tamamen yok edip yeniden kromozom oluşturan ***tufan etkisi*** yaklaşımları uygulanmıştır. Bu yaklaşımlar ile beraber D_A flip-flopu için elde edilen kombinasyonel lojik devre için başarı oranı 53/64' den 56/64' e yükselmiştir. Bu yaklaşımların uygulandığı ve uygulanmadığı GA için performans eğrileri Şekil 5.18. ve Şekil 5.19 da görüldüğü gibidir.

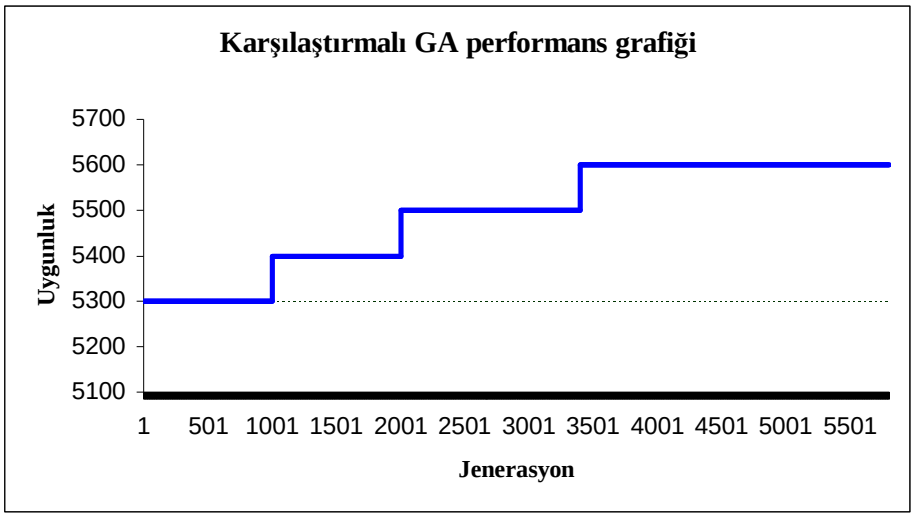


Şekil 5.18. Adaptif mutasyon ve tufan etkisi yaklaşımları uygulanmamış GA performans eğrisi



Şekil 5.19. Adaptif mutasyon ve tufan etkisi yaklaşımları uygulanmış GA performans eğrisi

Şekil 5.18 incelendiğinde 14196'ıncı jenerasyondan itibaren uygunluk değeri aynı kalmaktadır. Bu jenerasyondan itibaren *adaptif mutasyon* ve *tufan etkisi* yaklaşımları Şekil 5.19'da görüldüğü gibi etkin olmaktadır. 14196'ıncı jenerasyon öncesinde uygunluk değeri 1000 jenerasyon boyunca aynı kalmadığından bu yaklaşımlar etkin olmamaktadır. Dolayısıyla 14196'ıncı jenerasyona kadar performans eğrileri Şekil 5.18 ve Şekil 5.19'da paralellik göstermektedir. GA'nın14196'ıncı jenerasyon sonrasındaki *adaptif mutasyon* ve *tufan etkisi* yaklaşımları aktif ve pasif durumda iken performans eğrileri karşılaştırmalı olarak Şekil 5.20'de görülmektedir.



Şekil 5.20. Karşılaştırmalı GA performans grafiği

D_B için elde edilen kombinasyonel lojik devrede ise tutmayan 2 durum söz konusudur. Bu durumda başarı 62/64 yani %96,8 olmaktadır. Son olarak çıkış için elde edilen lojik devrede tutmayan 1 durum söz konusudur ve GA'nın başarısı 63/64 yani %98,4 olmaktadır.

Devrenin geneli için bakıldığında toplamda 10 değer tutmayıp GA için genel başarı oranı 54/64 yani %84,3 olmaktadır.

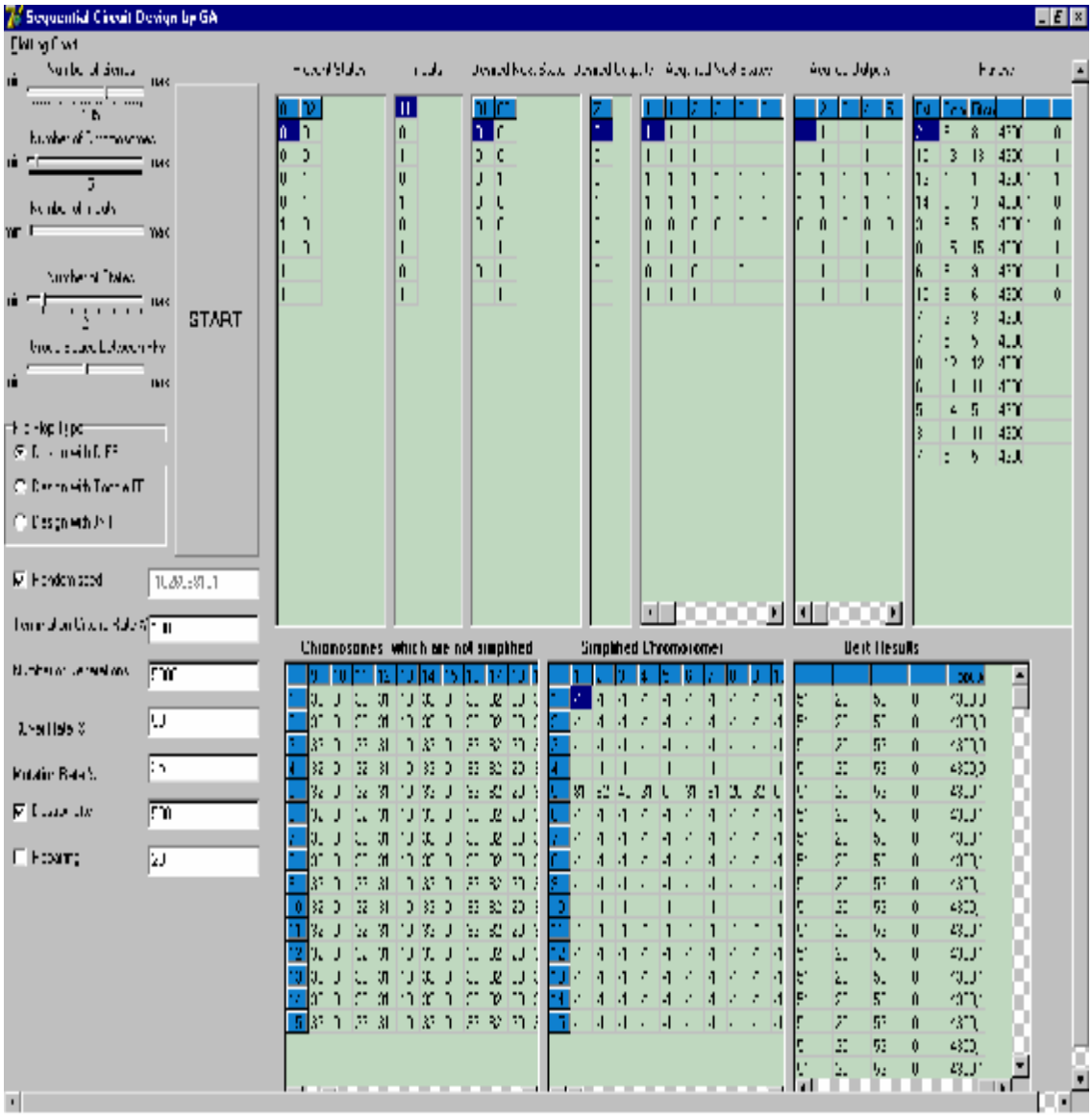
5.4. GA Test sonuçları

Çalışmamızda çeşitli sayıda girişlere sahip 2 durumlu ardışık lojik devreler GA ile tasarlanmıştır. Bu tasarımlar elde edilirken farklı sayıda çözüm toplulukları, farklı değerde çaprazlama ve mutasyon oranları kullanılmıştır. Bu tasarımlar elde edilirken 2 durumlu ve 1 girişli ile 2 durumlu ve 2 girişli ardışık lojik devrelerde GA tarafından yapılan tasarımlar elde edilmek istenen çıkış ve sonraki durum değerlerini birebir sağlarken 2 durumlu ve 4 girişli ardışık lojik devre tasarımında % 84,3'lük bir başarı oranı yakalayabilmiştir.

Tasarımların bazıları gerçekleştirilirken en iyi çözüme ait uygunluk değerinin 200 jenerasyon boyunca aynı kalması durumunda mutasyon oranı %1 oranında arttırılarak çözüm topluluğuna mutasyon ile çeşitlilik kazandırma ve çözüm topluluğunun tek düzeleşmesini engelleme amaçlanmıştır. Bunun yanı sıra uygunluk

değerinin 1000 jenerasyon boyunca aynı kalması halinde *tufan etkisi* oluşturularak var olan çözüm topluluğu tamamen yok edilip yerine yeni bir çözüm topluluğu oluşturma yaklaşımları kullanılmıştır.

Ardışık lojik devre tasarımı için GA kullanan programın ekran çıktısı Şekil 5.18' de görüldüğü gibidir.



Şekil 5.21. GA için geliştirilen programın ekran çıktısı

Şekil 5.20' den görülebileceği gibi, ardışık lojik devre tasarımı için geliştirilen programda her bir kromozoma ait gen sayısı, popülasyona ait kromozom sayısı, ardışık lojik devreye ait giriş sayısı, ardışık lojik devrede bulunacak flip-flop sayısı, toplam

jenerasyon sayısı, elde edilecek devrenin yüzdelik olarak sağlanması istenen doğruluk oranı, çaprazlama oranı, mutasyon oranı, *tufan etkisi* yaklaşımının kaç jenerasyonda bir gerçekleşmesi gerektiği ayarlanabilmektedir. Bunların yanı sıra, elde edilemek istenen ve gerçekte elde edilen ardışık lojik devrelerin doğruluk tabloları, iki parametreli turnuva metodunda seçilen iki kromozomdan hangisinin daha uygun olduğu, geri yayılımı yapılmış ve yapılmamış kromozomlara ait içerikler, her bir jenerasyondaki popülasyona ait en uygun kromozomun içeriği ve uygunluğu yer almaktadır. Ayrıca, her jenerasyonda elde edilen en uygun kromozoma ait uygunluk değerinin grafik içerisinde takibi sağlanarak GA ya ait performans grafiği oluşturulabilmektedir.

6. SONUÇ VE ÖNERİLER

Bu tezde Evrimleşebilen Donanım (**E**volvable **H**ard**W**are-EHW) konusu ve uygulama alanları literatürden incelenerek, kullanımı ile ilgili örnek çözümler ve bunlara ait sonuçlar sunulmuştur.

Tezde, EHW'nin **harici** uygulamaları için geliştirilen Genetik Algoritma (GA) ile kombinasyonel lojik devre ve hafıza elemanından oluşan belirli sayıdaki girişe ve çıkışa sahip ardışık lojik devre tasarımı ve optimizasyonu amaçlanmıştır.

Öncelikle hedef devreye ait durum tablosu veya durum diyagramından yararlanarak, devreye uygulanan girişler, hafıza elemanlarındaki önceki durumdan sonraki duruma geçiş sırasındaki durum değişiklikleri ve bu değişikliklerin devre çıkışına olan etkisi gibi, veriler GA'ya aktarılmıştır. Bu veriler doğrultusunda GA belli sayıda çözüm topluluğu oluşturmuş ve bu çözüm topluluğuna genetik operatörler algoritmadaki sırasıyla uygulanarak sonlandırma kriterine göre GA işleyişi tamamlanmıştır. Elde edilen çözümler elektronik simülasyon programı ile denenmiş ve sonuçları alınmıştır.

Deneyler için seçilen hedef devrelerin bir kısmında istenen fonksiyonun %100'ü sağlanırken, bir kısmında yaklaşık sonuçlar elde edilmiştir. 1 girişli 2 durumlu ve 2 girişli 2 durumlu ardışık lojik devre tasarımlarında durum tablosu %100 başarı ile elde edilirken, 4 girişli 2 durumlu ardışık lojik devre tasarımı için standart GA ile %82.8'lik başarı elde edilmiştir. Bu başarı oranını arttırma amacıyla, çözüm topluluğunun tek düzeleşmesi halinde bu durumu engellemek için *adaptif mutasyon*, çözüm topluluğunu yeniden oluşturmak için *tufan etkisi* ve çözüm arama sınırlarını daraltmak amacıyla devrenin bir kısmı için *devre yoğunlaşma* gibi soruna özel yaklaşımlar kullanılmıştır. *Adaptif mutasyon* ve *tufan etkisi* yaklaşımları ile başarı oranı kısmi kombinasyonel devre için % 82.8 den % 87.5' e yükselmiştir. Ayrıca GA tarafından tasarlanan devrelerde, klasik tasarımdan farklı sonuçlar elde edilmiş ve tasarlanan devrelerdeki kapı sayısı azaldığı gözlenmiştir.

Yapılan bu çalışmalar ışığında, EHW ile ardışık lojik devre tasarımında başarı oranını arttırmak amacıyla,

- Kullanılan elektronik eleman kütüphanesinin içeriği genişletilerek elde edilecek yeni çözümlerle karşılaştırılması,
- Mevcut ticari çizim program eleman kütüphanelerinin, geliştirilen GA ile kullanılıp devre tasarımlarının incelenmesi,
- Gerçek zamanlı EHW robot uygulaması yapılması,

öngörülen çalışmalar arasındadır.

KAYNAKLAR

ABD-EL-BARR, M., SAIT, S. M., SARIF, B. A. B., AL-SAIARI, U., 2003. A Modified Ant colony Algorithm for Evolutionary Design of Digital Circuits, **Evolutionary Computation, 2003. CEC '03. The 2003 Congress on**, pp. 708-715 Vol.1.

AKSOY, L., 2004. "Design of Sequential Circuits Using Genetic Programming" **http://www3.itu.edu.tr/~okerol/Sequential%20Circuit%20Design.pdf.**

ALBERT, D., 1997. "Evolutionary Hardware Overview" **http://citeseer.ist.psu.edu/albert97evolutionary.html.**

ALI, B., ALMAINI, A.E.A., KALGANOVA, T., 2004. Evolutionary Algorithms and Their Use in the Design of Sequential Logic Circuits, **Int. journal on Genetic Programming and Evolvable Machines, Kluwer Academic Publishers, Vol. 5 No.1 pp.11-29.**

AMARAL, J. N., 2003. Quine McCluskey Method. Lecture Notes of Computer Organization and Architecture II. Department of Computing Science University of Alberta. November 28, 2003. **http://www.cs.ualberta.ca/~amaral/courses/329/webslides/Topic5-QuineMcCluskey/sld003.htm.**

ANDERSEN, P., 1998. Evolvable Hardware : Artificial Evolution of Hardware Circuits in simulation and reality. Master's Thesis Computer Science. 15th May 1998 **http://www.boelskifte-andersen.dk/peter/th.ps.zip.**

ANONYMOUS, 2005. Sequential Circuit Analysis. **http://www.cs.uiuc.edu/class/fa05/cs231/lectures/14-SequentialAnalysis.pdf.**

ANONYMOUS, 2006. Application-specific integrated circuit. **http://en.wikipedia.org/wiki/Application-specific_integrated_circuit.**

ANISSIMOV, M., 2007. "What is Evolvable Hardware". **http://www.wisegeek.com/what-is-evolvable-hardware.htm.**

ÇAKIR, M., OZDEMIR, E., 2006. "Otoyol Aydınlatma Verimliliğinin Genetik Algoritma Kullanarak Optimizasyonu" 6. Ulusal Aydınlatma Kongresi –İTÜ Taşkışla Binası-Kasım, 2006.

GOLDBERG, D. E., 1989. **Genetic Algorithms in Search, Optimization, and Machine Learning**, Addison-Wesley Publishing Comp., ISBN 020-115-767-5.

GORDON, T. and BENTLEY, P.**,** 2002 On Evolvable Hardware. **In Soft Computing in Industrial Electronics, S. Ovaska and L. Sztandera, Eds. Heidelberg, Germany.: Physica-Verlag, 2002. http://citeseer.ist.psu.edu/gordon02evolvable.html.**

GUDISE, V.G., VENAYAGAMOORTHY, G.K., 2003. Evolving Digital Circuits Using Particle Swarm, **Neural Networks, 2003. Proceedings of the International Joint Conference on**, pp. 468-472.

HIGUCHI, T., IBA, H., MANDERICK, B. 1994. Massively parallel artificial intelligence. pp. 398-421.

HIGUCHI, T., IWATA, M., KEYMEULEN, D., SAKANASHI, H., MURAKAWA, M., KAJITANI, I., TAKAHASHI, E., TODA, K, SALAMI, M., KAJIHARA, N., OTSU, N., 1999. Real-World Applications of Analog and Digital Evolvable Hardware. **IEEE Transactions on Evolutionary Computation, Vol. 3, No. 3, September 1999.**

HIGUCHI, T., IWATA, M., SAKANASHI, H., TAKAHASHI, E., MURAKAWA, M., KAJITANI, I., 2000. Dynamic Adaptive Devices and Their Application.

HIGUCHI, T., IWATA, M., KAJITANI, I., MURAKAWA, M., YOSHIZAWA, S. and FURUYA, T., 1996. "Hardware Evolution at Gate and Function Levels," in **Proceedings of Biologically Inspired Autonomous Systems: Computation, Cognition and Action**, Durham, North Carolina, March 1996.

HIRST, A J, 1997. Notes on The Evolution of Adaptive Hardware, HCRL, Open University, UK.

HOUNSEL, I., ARSLAN T. A Novel Genetic Algorithms for the Automated Design of Performance Driven Digital Circuits.

KALAYCI, T.E., 2006. Genetik Algoritmalar. **http://kodveus.blogspot.com/2006/09/genetik-algoritmalar-blm-iv-seim.html.**

KARABOĞA, D., 2004. **Yapay Zeka Optimizasyon Algoritmaları**. Atlas yayın dağıtım, Yayın no:38, 199s, İstanbul.

KERT, M., 2006. Gerçek Görüntüden Elde Edilen Koordinatlarla Robot Kol Hareket Optimizasyonu. Yüksek Lisans Tezi.

KOZA, J.R., 1995. Survey of Genetic Algorithms and Genetic Programming. In Proceedings of the Wescon 95 **- Conference Record: Microelectronics, Communications Technology, Producing Quality Products, Mobile and Portable Power, Emerging Technologies,** San Francisco, CA, 7.-9. November 1995. IEEE, New York, NY. **http://citeseer.ist.psu.edu/koza95survey.html.**

LINDGREN, P., 1999. Applications of Decision Diagrams in Digital Circuit Design. **http://www.ee.pdx.edu/~mperkows/PerkowskiGoogle/thesis-Lindgren.pdf.**

MANDERICK, B., HIGUCHI T., 1996. Recent Advances in Evolvable Systems, **ICES 96 (International Conference on Evolvable Systems)**. Tsukuba, Japan.

MILLER, J.F., JOB, D.,VASSILEV, V. K., 2000. Principles in the Evolutionary Design of Digital Circuits Part I. **http://citeseer.ist.psu.edu/miller00principles.html.**

MANARESI, N., FRANCHI, E., GUERRIERI, R., BACCARANI, G., 1996. A field Programmable Analog Fuzzy Processor with Enhanced Temperature Performance. **Proc. ESSCIRC'96**, pp. 152-155, Neuchatel 1996. **http://citeseer.ist.psu.edu/363155.html.**

NABİYEV, V. V., 2003. **Yapay Zeka.** Seçkin Yayıncılık. 724 s, Ankara.

OBITKO, M., 1998. Introduction to Genetic Algorithms. Czech Technical University **http://cs.felk.cvut.cz/~xobitko/ga.**

PERKOWSKI, M. 1986, Digital Design Automation: Finite State Machine Design. **http://web.cecs.pdx.edu/~mperkows/=FSM/finite-sm/finite-sm.html.**

POPA, R., 1999. Evolutionary Design of Digital Circuits-An Alternate Approach to the Conventional Logic Design, **The Annals of "Dunarea De Jos" University of Galati Fascicle III,** Romania.

ROGGEN, D., 2005. Multi-Cellular Reconfigurable Circuits: Evolution, Morphogenesis and Learning. **http://citeseer.ist.psu.edu/roggen05multicellular.html.**

SAKANASHI, H., IWATA, M., KEYMULEN, D., MURAKAWA, M., KAJİTANİ, I., TANAKA, M., HIGUCHI, T., 1999. Evolvable Hardware Chips and their Applications, **Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on.** pp**:**559-564 vol.5, Tokyo, Japan.

STROUD, C.E., 2006. Sequential Logic Analysis. **www.eng.auburn.edu/~strouce/class/elec2200/elec2200-11.pdf.**

STOMEO, E., KALGANOVA, T., LAMBERT, C., LIPNITSAKYA, N., YATSKEVICH, Y., 2005. On Evolution of Relatively Large Combinational Logic Circuits, **Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on.**

STOICA, A., ZEBULUM, R., KEYMEULEN, D., JIN, Y., DAUD, T,. 2000. **Smart Engineering Design (ANNIE 2000)** Rolla, Missouri, USA.

TAN, K.C., WANG, L.F., LEE, T.H.,VADAKKEPAT, P., 2004. Evolvable Hardware in Evolutionary Robotics. **http://www.springerlink.com/index/X2342GP47783642V.pdf.**

TEKTAŞ, M., AKBAŞ, A., TOPUZ, V., 2007. "Yapay Zeka Tekniklerinin Trafik Kontrolünde Kullanılması Üzerine Bir İnceleme". **http://otvav.wordpress.com/tag/teknoloji-teknik-ilim.**

THOMPSON, A., 1996. Silicon Hardware**, Proceedings of Genetic Programming MIT Press 1996 (GP96)**, pp. 444-452.

TORRESEN, J., 2004. An Evolvable Hardware Tutorial. **http://folk.uio.no/jimtoer/fpl04_Torresen.pdf.**

TYRRELL, A.M., HOLLINGWORTH, G., SMITH, S.L. 2001. Evolutionary Strategies and Intrinsic Fault Tolerance, **Evolvable Hardware, 2001. Proceedings. The Third NASA/DoD Workshop on**, Long Beach, CA, USA.

WHITLEY, D., 1994. A Genetic Algorithm Tutorial. Statistics and Computing Volume 4 Sayfa 65-85, 1994. **http://citeseer.ist.psu.edu/367462.html.**

WOO, J. W., PARK, T.S., LEE, H.L., 1999. Adaptive Hardware Evolution under Unpredictable Environmental Changes, **AP-ASIC '99 The First IEEE Asia Pacific Conference**, Seoul, Korea, pp. 372-375.

YAO, X., HIGUCHI, T., 1999. Promises and Challenges of Evolvable Hardware. **IEEE Transactions on Systems, Man, and Cybernetıcs-Part C: Applications and reviews, vol. 29, no. 1**, pp. 87-97, February 1999.

ZEBULUM, R.S., AURELIO P. M. and VELLASCO, M., 1996, Evolvable Systems in Hardware Design: Taxonomy, Survey and Applications, **Proc. of the 1st Int. Conf. on Evolvable Systems**, Tsukuba, Japan, pp. 344-358.

ZHOU, H., 2002. Advanced Digital Design. Lecture Notes 3 : Two-Level Logic Minimization Algorithms. **http://www.ece.northwestern.edu:80/~haizhou/Lec03.ppt.**

ÖZGEÇMİŞ

1979 İskenderun'da doğdum. İlk, orta ve lise öğrenimimi İskenderun'da tamamladım. 1997 yılında Pamukkale Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği bölümünü kazandım ve 2001 yılında mezun oldum. 2004 yılında Mustafa Kemal Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Ana Bilim Dalında yüksek lisansa başladım. 2004 yılından bu yana Mustafa Kemal Üniversitesi İskenderun Meslek Yüksekokulu Elektronik Haberleşme Programında Öğretim Görevlisi olarak görev yapmaktayım. Evliyim.