KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ HABERLEŞME SİSTEMLERİNDE FPGA UYGULAMASI

YÜKSEK LİSANS

Elektronik ve Haberleşme Müh. Şener DİKMEŞE

Anabilim Dalı: Elektronik ve Haberleşme Mühendisliği

Danışman: Prof. Dr. Hasan DİNÇER

KOCAELİ, 2007

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ HABERLEŞME SİSTEMLERİNDE FPGA UYGULAMASI

YÜKSEK LİSANS TEZİ

Elektronik ve Haberleşme Müh. Şener DİKMEŞE

Tezin Enstitüye Verildiği Tarih: 04 Haziran 2007

Tezin Savunulduğu Tarih: 23 Temmuz 2007

Tez Danışmanı	Üye	Üye
Prof.Dr.Hasan DİNÇER	Yrd.Doc.Dr.Sıtkı Öztürk	Yrd.Doç.Dr.Soner ÖZGÜNEL
(.....)	(.....)	(.....)

KOCAELİ,2007

ÖNSÖZ

Günümüzde Kablosuz Haberleşme Sistemleri kullanımı yaygın görülmektedir. Özellikle Gezgin Haberleşme Sistemleri hayatımızın önemli bir parçası haline gelmiştir. Gezgin haberleşme sistemlerinin bir zorunluluk haline gelmesi bu sistemlerin sayısını her geçen gün arttırmaktadır. Sayının artmasının yanında kullanıcıların istekleri de atrmaktadır. Fakat kullanılabilecek frekans bölgesinin sınırlı olması operatörleri çeşitli araştırmalar yapmaya itmiştir. Son yıllarda çalışmalar kapasite arttırımı için önemli avantajlara sahip Akıllı Anten Sistemlerine odaklanmıştır. Bu sistemler yalnızca istenen işaret doğrultusunda örüntü oluşturacağından, sistem kapasitesi büyük ölçüde artmış olacaktır. Fakat istenen işaretin yakalanabilmesi için gelişmiş işaret mekanizmalarının kullanılması gerekir.

Yapılan çalışmada istenen işaret doğrultusunda işaret örüntüsü oluşturabilmek amacıyla çeşitli akıllı anten algoritmaları Alan Programlamalı Kapı Dizileri (Field Programmable Gate Arrays, FPGA) kullanılarak gerçeklenmiştir

Çalışmalarım sırasında bilgi ve birikimleri ile her zaman bana destek olan danışman hocalarım Sayın Prof. Dr. Hasan DİNÇER ve Sayın Doç. Dr Adnan Kavak'a, değerli bilgi ve birikimleriyle benden yardımlarını esirgemeyen hocalarım Öğr. Gör Suhap Şahin, Araş. Gör Kerem Küçük, Araş. Gör Halil Yiğit ve Sayın Mustafa Karakoç'a teşekkürü bir borç bilirim.

Aynı zamanda 6 aylık bir çalışma için beni South Florida Üniversitesi'ne davet eden gerçek manada akademisyenliği öğreten değerli hocam Prof. Dr Hüseyin Arslan ve beni kendi grubundan gibi gören yardımlarını esirgemeyen Kablosuz Haberleşme ve İşaret İşleme Grubu'nun (Wireless Communication and Signal Processing) her üyesine sonsuz şükranlarımı sunarım.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER DİZİNİ VE KISALTMALAR LİSTESİ

Simgeler

$a(\theta)$	: yönlendirme vektörü
c	: ışığın boşlukta yayılma hızı
c(k,t)	: PN dizisi
d	: elementler arası uzaklık
d(k,t)	: trafik ve pilot bilgi işareti
ε(t)	: hata işareti
f	: frekans
I(t)	: girişim işareti
M	: element sayısı
$n(t)$	: gürültü işareti
$s(t)$	: giriş işareti
T_s	: örnekleme hızı
v	: dalga hızı
w	: ağırlık fonksiyonu
λ	: dalga boyu
θ	: geliş açısı (geliş doğrultusu)
$x(t)$	: anten elementlerinde alınan işaret
$x(n)$	: örneklenmiş işaret
$y(t)$	: çıkış işareti
$z(k)$	: beamformer çıkışı
3G	: Third Generation (3. nesil)

Kısaltmalar

ADC	: Analog Digital Converter (analog sayısal dönüştürücü)
AGF	: Alçak Geçiren Filtre
ASIC	: Applicable Specific Integrated Circuit (genel amaçlı işlemci)
BER	: Bit Error Rate (bit hata oranı)
CDMA	: CodeDivision Multiple Access (kod bölmeli çoklu erişim)
CM	:Constant Modulus (sabit modül)

CPLD	:Complex Programmable Logic Devices
DAC	:Digital Analog Converter
DSP	: Digital Signal Processor (sayısal işaret işlemcisi)
DOA	: Direction Of Arrivals (geliş açısı doğrultusu)
FDD	: Frequency Division Dublex
FDMA	: Frequency Division Multiple Access (frekans bölmeli çoklu erişim)
FPGA	:Field Programmable Gate Arrays (Alan Programlamalı Kapı Dizileri)
IF	: Intermediate Frequency (ara frekans)
LMS	: Least Mean Squares (en küçük kareler yöntemi)
LNA	:Low Noise Amplifier (Düşük Gürültü Yükselteci)
LS	: Least Squares (En Küçük Kareler)
MMSE	: Minimum Mean Square Error (en küçük ortalama karesel hata)
MRC	: Maximal Ratio Combining (en büyük oran birleşimi)
MSE	: Mean Square Error (ortalama karesel hata)
MVDR	: Minimum Variance Distortionless Response
MUSIC	: Multiple Signal Classification (çoklu işaret sınıflandırılması)
ESPRIT	: Estimation of Signal Parameters via Rotational Invariance techniques
RF	: Radio Frequency (Radyo frekansı)
RLS	: Recursive Least Squares (rekürsif en küçük kareler yöntemi)
SAS	:Smart Antenna System
SCC	:Space Cod Correlator (uzay kod korelatör)
SDR	:Software Defined Radio
SDMA	: Spatial Division Multiple Access (uzamsal bölmeli çoklu erişim)
SDR	: Software Defined Radio (yazılım radyo)
SINR	: Signal Interference Noise Ratio (işaret gürültü ve etkileşim oranı)
SIR	: Signal Interference Ratio (işaret etkileşim oranı)
SNR	: Signal to Noise Ratio (işaret gürültü oranı)
TDD	: Time Division Dublex (zaman bölüşümlü çoğullama)
TDMA	: Time Division Multiple Access (zaman bölmeli çoklu erişim)
ULA	:Uniform Linear Array (Düzgün Doğrusal Dizi)
UMTS	: Universal Mobile Telecommunications System
VHDL	: Very High Speed Integrated Description Language

KABLOSUZ HABERLEŞME SİSTEMLERİNDE FPGA UYGULAMASI

Şener DİKMEŞE

Anahtar kelimeler : Akıllı Anten Sistemi, 3.Nesil, Işın Şekillendirme, Akıllı Anten Algoritmaları, FPGA ve VHDL

Özet : Son yıllarda Kablosuz haberleşme Sistemleri için Gezgin Haberleşme Sistemleri tüm dünyada büyük ilgi gördü. Akıllı Anten Sistemleri Kablosuz Haberleşme Sistemleri içinde etkili spektrum kullanarak kapasite arttırımı için anahtar bir teknoloji olarak farkedildi.

Araştırmalar son zamanlarda gelişmiş Yazılım Tanımlı Radyo (SDR), anten sistemleri ve işaret işleme tekniklerine odaklandı. Radyo olarak tanıtılan SDR yazılımla birlikte çok etkili bir maliyetle yenilenebilir ve radyo nimarisini geliştirmek için tercih edilebilir.

3. Nesil CDMA temel istasyonu, Akıllı Anten Sistemi Frekans Bölüşümlü Çiftleme (Frequency Division Dublex, FDD) için akıllı anten algoritmalarına gerek duyar. Küçük hesaplama karmaşıklığı ile istenen performans kolaylıkla sağlanabilir. Programlanabilir işlemciler kullanıldığı zaman bu algoritmalar akıllı anten temel istasyonu içinde 3. Nesil temel istasyon içinde kolayca güncellenir. Alan Programlamalı Kapı Dizileri (FPGA) çok fonksiyonlu yazılım tanımlı radyo uygulamaları gibi kablosuz haberleşme sistemlerinde oldukça önemli bir yere sahiptir. FPGA teknolojisi ile gerçek zamanlı uygulamalara yakın sonuçlar elde edilebilir. İşaret işleme ise lojik tasarım birleştirilmesiyle birlikte esnek bir yapıya sahiptir. Bu nedenle donanım kullanıcıları DSP' ler ile yer değiştirmektedir.

FPGA uygulaması için Modelsim gibi bir VHDL aracı kullanılarak adaptif algoritmalara uygulandı. Temel istasyonda anten dizileri için Doğrusal Anten Dizisi düşünülür. Anten dizisinde alınan işaretin cdma2000 formatında iletildiği ve de çokluyolun işareti yavaşlattığı farz edilir. Burada kullanılan Uzay Kod Korelatör algoritmasının avantajı En Küçük Kareler, Yenilenebilir En Küçük kareler ve Sabit Modül algoritmasına benzememesidir. Bu algoritma eğitim parametreleri gerektirmez ve ağırlık vektörü hesaplama zamanı dizi topolojisi ve çoklu yol yayılım durumlarından etkilenmez.

IMPLEMENTATION FPGA OF WIRELESS COMMUNICATION SYSTEMS

Şener DİKMEŞE

Keywords : Smart Antenna System, 3G, Beamforming, Smart Antenna Algorithms, FPGA and VHDL

Abstract : Mobile Communication Systems for Wireless Communications Systems has received enormous interest worldwide in recent years. And smart antenna systems are recognized as a key technology for increasing capacity and using efficient spectrum in wireless communication systems.

Researches have recently been focusing especially on advanced software radio, antenna systems and signal processing techniques. Software Defined Radio (SDR) is often described as a radio can be upgraded very cost effectively with software and offers the ability to develop radio architecture.

Integrating on of SAS into 3G CDMA base stations requires smart antenna algorithms that are suitable for Frequency Division Dublex (FDD) mode, have small computational complexity, and provide desired performance. When implemented on programmable processors, these algorithms will facilitate upgrade of existing 3G base stations into smart antenna capable base station.Field Programmable Gate Array (FPGA) is important in multi function software defined radio applications like wireless communication. Developments in FPGA technology have enabled near real-time signal processing. The flexibility of having the ability to integrate logic design with signal processing is showing hardware designers to replace digital signal processor with FPGAs.

For the FPGA implementation, adaptive algorithms were implemented by using VHDL tool such as Modelsim. As the antenna arrays at the base station, uniform linear array (ULA) topology is considered. Signal received from the antenna array is assumed to be transmitted in cdma2000 format from the mobile terminal and exposed to changing multipath conditions. The advantage of using Space Code Corelator (SCC) algorithm is that unlike other adaptive algorithms such as Least Mean Square (LMS), Recurisive Mean Square (RLS) and Constant Modulus (CM) algorithms, it does not require any learning parameter and that its weight vector computation time is not affected by the array topology and multipath propagation conditions.

BÖLÜM 1. GİRİŞ

1.1 Kablosuz Haberleşme Sistemleri

Kablosuz haberleşme sistemlerinin kullanımı her geçen gün artmaktadır. Bu yüzden kablosuz haberleşme sistemleri günümüzde hayatımızın önemli bir bölümünü kapsamaktadır. Hem kişisel hem de iş çalışmalarında kablosuz haberleşme sistemlerinin kullanımı mevcuttur. Bu kullanımın artması yanında birçok problemi de getirmektedir. Çünkü mevcut kapasite günümüzde artan kullanıcı sayısını karşılamadığı gibi gelecekte bu problemler artarak devam edecektir [1].

Özellikle gezgin haberleşme sistemleri, kablosuz haberleşme sistemlerinin önemli bir bölümünü oluşturuyor. Kullanıcı sayısı her geçen gün artmaktadır. Bu sayı milyarları geçmiştir. Bu artan sayı, operatörler için her ne kadar güzel gibi gözükse de bu noktada operatörlere büyük işler düşmektedir. Çünkü, operatörler kullanıcının isteklerini karşılamanın yanında ortam şartlarının da haberleşme için uygun olmasını sağlamalıdır.

Kullanıcı sayısının artması çoğu zaman yetersiz kapasiteden dolayı haberleşmenin aksamasına veya kesilmesine sebep olur. Bu da hem kullanıcı hem de operatörler tarafından istenmeyen durumdur. Bu yüzden artan kullanıcı sayısından dolayı gelecek nesil kablosuz haberleşme sistemlerinde sistem performansı ve kullanıcı istekleri önem kazanmıştır. Araştırmacılar bu problemlerin çözülmesi için çesitli alternatif düşünceler ortaya atmışlardır. Bunlar kısaca maddeler halinde açıklanacak.

1. Band genişliğini arttırmak: Artan kapasite ihtiyacını karşılamak için her ne kadar çözüm gibi gözükse de kullanılan frekans bandının sınırlı olması ve bu frekans bandının arttırılmasının yüksek maliyet gerektirmesi nedeniyle optimum bir çözüm sağlamaz.

2. Mevcut olan hücrelerin daha küçük hücrelere bölünmesi. Bu da sistem maliyetini aşırı arttıracağı için kullanılamaz.

3. Düşük oranlı kodek kullanımı hareketlilik çok fazla oranda olabileceği için çok iyi performans sağlamaz.

4. Alıcı hassasiyeti çok iyi özelliklere sahip işlemcilerle arttırılabilir fakat bu da maliyeti önemli ölçüde arttıracaktır.

Yukarıda ifade edilen çözümler maliyet işlem yükü gibi çeşitli sebeplerden dolayı tercih edilmez. Aynı zamanda kablosuz haberleşme sistemleri ile kablolu haberleşme sistemlerini karşılaştırdığımızda performans analizi açısından çok büyük farklar bulunmaktadır. Özellikle kablosuz haberleşme sistemlerinde işaret kanaldan geçerken çeşitli girişim, gürültü, dağ, vadi ve binalar gibi fiziki ortam şartlarından dolayı çokluyollara maruz kalabilir. Bu, sistemin performansını olumsuz yönde etkileyecektir.

Bu sebeple yukarıda ifade edilen kötü şartların ortadan kaldırılabilmesiyle işaret daha iyi bir şekilde alınarak, haberleşmenin daha sağlıklı yapılması sağlanacaktır.

Kapasite sıkıntısı ve de kanalın olumsuz etkisi, aklımıza anten mimarisini getirir. Çünkü, anten mimarisinin uygun kullanımı işaretin gereksiz yayılmasını engelleyeceği gibi sistem performansını etkileyen girişimler için de etkili çözüm sağlar. Bir de yalnızca işaret doğrultusunda örüntü oluşturulacağı için çokluyollar içinde iyi bir çözümdür [2].

Bu sebeple yapılan bu çalışmada gelecek nesil kablosuz haberleşme algoritmalarından özellikle akıllı anten algoritmalarından bahsedilecek. İşlemci fiyatlarının düşmesi ile beraber, akıllı antenlerin kullanımı yukarıda sayılan sebeplerden dolayı artmaktadır. Bu yüzden Sayısal İşaret İşlemci (Digital Signal Processing, DSP) ve Alan Programlamalı Kapı Dizileri (Field Programmable Gate Array, FPGA) kullanılarak uygun algoritmalar ile istenen işaret doğrultusunda örüntü, mikro ve mili saniyeler mertebesinde bulunabilir. Gezgin haberleşme

sistemleri genellikle hareket halinde bulunabileceğinden dolayı algoritmaların çok kısa sürede işlem yapması gerekir. Özellikle yapılan çalışmada kullanılan FPGA'ler hız açısından ve istenen işareti yakalama performansı açısından iyi sonuçlar ortaya koymaktadır [3].

1.2 Akıllı Antenlere Genel Bir Bakış

Genel anlamda, Akıllı Anten terimi, sabit ortagonal ışın demetli ve sınırlı zekaya sahip bazı dizilerden, karmaşık yapı ve kontrol algoritmalarıyla oldukça karmaşıklaşan adaptif dizilere kadar antenlerin büyük bir kısmını kapsayabilir.

Kesin olan kanı, bu terimin, birbirine benzeyen antenlerden oluşan basit bir diziden çok fazlasını ifade ettiğidir. Birden fazla örüntüye sahip basit bir dizinin, akıllı anten kapsamına girebilmesi için, istenen işaretin geldiği yöne göre örüntüyü seçebilecek, düşünüp karar verme mekanizmasına sahip olması gerekir.

Günümüzde mevcut bulunan geleneksel anten dizilerinde ana örüntü istenen işaret doğrultusunda yönlendirilir. Bu faz dizisi, ışın yönlendirilmiş diziler veya taranmış diziler olarak adlandırılır. Faz kaymaları nedeniyle yönlendirilen bu örüntü istenen işaretin doğru bir şekilde alınmasını sağlar.

Modern ışın şekillendirme dizileri, örüntünün istenen işaret doğrultusunda kriter değerlere göre şekillendirilmesini sağlar. Akıllı antenler alternatif olarak tam olarak aynı anlama gelmemelerine rağmen sayısal ışın biçimlendirme (digital beamforming, DBF) veya uyarlanabilir diziler (adaptive arrays) olarak da adlandırılır [4].

Akıllı terimi, belirlenen istenen işaret doğrultusundaki örüntünün işlenmesi anlamını ifade eder. Akıllı genellikle anten performansının hesaplama kontrolü anlamını ifade eder.

Akıllı antenlerin, geliştirilmiş radar sistemleri ve gezgin haberleşme sistemlerinde uzay bölüşümlü çoklu erişim (spatial division multiple access, SDMA) ile kullanımı önerilir.

Akıllı anten örüntüsü belirli algoritmalar yoluyla kriter değere göre şekillendirilir. Doğru kriterin belirlenmesi, işaret girişiminin (signal to interference, SIR) maksimize edilmesi ve ortalama karesel hatanın (mean square error, MSE) minimize edilmesi anlamına gelmektedir [5].

Böylece yalnızca istenen işaret doğrultusunda örüntü oluşturularak istenmeyen girişimler doğrultusunda boş örüntüler ile yüksek performans sağlanır. Bu sebeple işaret işleme birimleri ile enerjinin boşa harcanmasının önüne geçilerek dolaylı yoldan girişimler engellenmiş olur. Daha gelişmiş sistemler ile girişim doğrultusunda boş örüntü oluşturulur.

Dizi çıkışlarının bir analog sayısal dönüştürücü (analog digital converter, ADC) kullanılarak sayısallaştırılması bir gerekliliktir. Çünkü günümüz teknolojisinde işaret işlemcilerin örnekleme frekansı sınırlı olduğundan bu işlemler gerçekleştirilir. Bu sayısallaştırma işlemi temel band veya orta frekans bandında (intermediate frequency band, IF) gerçekleştirilir.

Anten örüntüsü sayısal işaret işleme ile gerçekleştirildiğinden bu işlem sık sık Sayısal Işın Biçimlendirme (digital beamforming) olarak adlandırılır. Algoritmalar adaptif olduğu zaman bu işlem adaptif (uyarlanabilir) ışın şekillendirici olarak adlandırılır.
Adaptif ışın biçimlendirme genel olarak sayısal ışın şekillendirmenin alt kategorisi olarak düşünülebilir.

Sayısal ışın şekillendirme ayrıca radar sistemleri, sonar sistemler ve haberleşme sistemlerine uygulanabilir. Sayısal ışın şekillendirmenin asıl avantajı, faz kayması ve dizi anten ağırlıklarının donanım üzerinde değil de yazılımsal olarak ilgili sayısal datanın üzerinde yapılıyor olmasıdır.

Adaptif ışın şekillendirme genel olarak düşünüldüğünde daha etkili ve kullanışlı ışın biçimlendirme çözümü sunar. Çünkü sayısal ışın şekillendirme ile elektromanyetik

ortamın değişimine göre dinamik olarak dizi paternini optimize eden bir algoritma içerir.

Geleneksel anten sistemlerini incelediğimiz taktirde ortamda bulunan çokluyol ve de girişim etkileri işaretin işaret gürültü oranını (signal to noise, SNR) aşırı derecede azaltır. Bu etki adaptif anten sistemleri kullanılarak önemli ölçüde arttırılır.

Gelecekte kullanılacak temel istasyonlar, mevcut çoğullama teknikleri kullanabilecekleri gibi uzay bölüşümlü çoklu erişim (spatial division multiple access) ile de sistem kapasitesi için önemli artışlar sağlar.

Bölüm 2' de akıllı antenler geniş bir şekilde anlatılacaktır.

1.3 Akıllı Antenterin Tarihsel Gelişimi

Adaptive antenleri geliştirme teknikleri 50' li yılların sonunda başlamıştır. Bu kelime adaptif dizi olarak Van Atta tarafından ortaya atılmıştır. 1959 yılında (kendi fazlı) Self Phased dizi olarak tasarlanmıştır. Öz (self) faz dizilerde faz eşlenik (conjuge) temeline dayanan akıllı faz şeması kullanılmıştır [6].

Faz Kilitlemeli Döngü (pahse lock loop, PLL) tekli ışın tarama işlemi kullanan faz dizi sistemlerinden oluşmakta olup 1960' larda kullanılmaya başlanmıştır [7].

Adaptif Kenar Lobu iptali (adaptive side lobe cancellation) Hovel tarafından 1959' da önerilen sistem olup, bu teknikte girişim işaretleri için boş örüntü oluşturulmuş ve böylece işaret gürültü oranı arttırılmıştır. Genelleştirilen işaret gürültü oranı (SNR) maksimize edilerek adaptif ışın şekillendirme algoritması geliştirilmiştir [8].

Daha sonraki aşamalarda En Az Karesel Ortalama (least mean square, LMS) algoritması Widrow tarafından uygulanmıştır [9].

Daha sonraki dönemlerde araştırmalar öz değer ve öz vektörlerin bulunması için geliştirilmiştir. Korelasyon matrisinde zayıf işarete veya gürültüye karşılık gelen

işaretler küçük özdeğer ifadeleri ile ve güçlü işarete karşılık gelen değerler büyük değerlikli öz değer ve öz vektörler hesaplamaları ile bulunabilir. Daha net sonuçların elde edilebilmesi için geniş yakınsama zamanları kullanılır. Reed Mallet ve Brennen Örnek Matris Tersi (Sample Matrix Inversion, SMI) için 1974 yılında çalışmalar yapmıştır [10].

Adaptif diziler için sonraki büyük tamamlayıcı nitelikler ise dizi izlemek için İzgesel Tahmin Metodları (Spectral Estimation) uygulanmasıdır. Capon 1969 yılında Maksimum Likelihood (ML) kullanarak işaret girişim oranını maksimize etme çalışmalarında bulunmuştur [11].

Yakın günümüzde de akıllı anten ve adaptif dizi çalışmaları devam etmektedir. Yazılım tanımlı radyo (software defined radio) ve kavramsal radyo (cognitive radio) gelişmesiyle birlikte yapılan çalışmalar daha fazla önem kazanmaktadır.

Özellikle günümüzde Mitola (1995), yazılım radyo perspektifi için önemli çalışmalarda bulunmuştur. Yapmış olduğu çalışmalarda matematiksel hesaplamalar ile yazılım tanımlı radyonun haberleşme sistemi için gereklilik olduğu anlaşılmıştır [12].

Aynı şekilde Naquip ve Godara (1996) ışın şekillendirme algoritmaları üzerinde hesaplamalar yapmışlardır. Özellikle bu dönemde gerçekleştirilen anten algoritmaları günümüzde kullanılması planlanan algoritmalar olmuştur [13].

1999 yılında kör algoritmalar ve kör olmayan algoritmalar yapan Rapaport önemli çalışmalarda bulunmuştur.

1.4 Tezin İşleyişi

Yazılım tanımlı radyo örneği teşkil edecek akıllı anten algoritmalarının Alan Programlamalı Kapı Dizileri (Field Programmable Gate Array, FPGA) kullanılarak gerçekleştirilmesi tezi oluşturuyor. Yukarı bağlantı kanalında kör olmayan algoritma olan en az karesel ortalama (Least Mean Squares, LMS), kör algoritma olan sabit

modül algoritması (Constant Modulus, CM) ve uzaysal kod korelatör (Space Code Correlator - SCC) algoritmaları yapılan çalışmada gerçeklenmiştir.

Hızlı bir performansa sahip olduğundan dolayı Alan Programlamalı Kapı Dizileri tercih edilmiştir. Tezde Xilinx ailesine ait Virtex4 ve Virtex 2 FPGA'leri programlamak amacıyla Yüksek Hızlı Tümleşik Devre Donanım Tanımlama Dili (Very High Speed Integrated Circuit Hardware Description Language, VHDL) kullanılmıştır. Aynı zamanda yapılan çalışmaların kit üzerinde denemeden önce yüksek simülasyon özelliklerine sahip, Modelsim simülasyon programı kullanılmıştır.

Şimdi tezin içeriği hakkında bilgi verilecek.

Bölüm1' de teze giriş yapıldı. Tez hakkında genel bilgiler ve tarihsel gelişim sunuldu.

Bölüm2' de akıllı anten sistemleri hakkında genel bilgiler verilecektir olup, akıllı antenlerin genel yapısı ve çeşitleri hakkında da temel bilgiler ifade edilecektir.

Bölüm3' de kullanılan FPGA ve FPGA ile programlamak için kullandığımız VHDL hakkında bilgiler verilecek olup, son kısımda ise kullandığımız VirtexII FPGA mimarisi tanıtılacaktır.

Bölüm4' de işaret modeli, işaret şekillendirme ile ilgili matematiksel kavramlar ve akıllı anten algoritmaları hakkında bilgi verilecektir.

Bölüm5' de algoritmaların FPGA üzerine nasıl uygulandığı anlatılacaktır. Ve de algoritma sonuçlarının karşılaştırılması verilecektir.

Bölüm6 sonuç bölümünden oluşmaktadır. Bu bölümde yapılan çalışma genel olarak değerlendirilecek ve de ileride yapılması planlanan çalışmalar hakkında genel bilgiler aktarılacaktır.

Genel olarak yapılan çalışmada gelecek nesil kablosuz haberleşme sistemleri için akıllı anten algoritmalarının FPGA kullanılarak gerçeklenmesi üzerine odaklanılacaktır. Böylece hem akıllı anten sistemleri hem de akıllı anten algoritmaları anlatılacağı gibi yüksek hız kabiliyetine sahip olan FPGA hakkında geniş bilgi verilecektir. Aynı zamanda sistem performansı daha önceden başka bir çalışma olarak yapılan DSP sonuçları ile karşılaştırılacaktır.

BÖLÜM 2. AKILLI ANTEN SİSTEMLERİ

2.1 Giriş

Kablosuz haberleşme sistemleri, özellikle de gezgin haberleşme sistemlerinin günümüzde kullanımının arttığı herkes tarafından farkedilmektedir. Kullanıcı sayısı artması beraberinde kapasite problemlerini ortaya çıkartmaktadır. Kapasitenin kullanıcı sayısına paralel artışı söz konusu olmadığından, bu durum yeni teknolojilerin araştırılması gerekliliğini ortaya koymuştur. Bu sebeple, yapılan çalışmalar sonucunda akıllı anten sistemlerinin bu problemin üstesinden geleceği düşünülmüştür. Akıllı anten sistemi kullanılarak çok fazla kullanıcının sınırlı band genişliği kullanımı probleminin üstesinden gelinmiştir. Özellikle istenen kullanıcı yönünde ışın örüntüsü oluşturularak aynı kanalın birçok kullanıcı tarafından kullanılması sistem kapasitesinin büyük ölçüde artmasını sağlayacaktır. Bu şekilde kapasitenin uygun kullanımı hem daha fazla kullanıcının haberleşmesini hem de kullanıcılar arasındaki girişimlerin önlenmesini sağlayacaktır. Böylelikle, hem haberleşme kalitesi artacak hem de kullanıcıların daha sağlıklı bir şekilde haberleşmesi sağlanacaktır. Bu sebeplerden ötürü, yapılan araştırmalar akıllı anten sistemlerinin gerekliliğini ortaya koymuştur.

2.2 Akıllı Anten Sistemleri

Akıllı anten sistemleri, yalnızca istenen kullanıcı doğrultusunda örüntü oluşturarak istenmeyen girişimlerin engellenmesini sağlayan karmaşık yapıya sahip bir sistemdir. Geleneksel anten dizilerinde, asıl ışın örüntüsü ilgili doğrultuya yönelir. Bu günümüzde 120°’ lik makro sektörler ile sağlanıyordu. Faz kaymaları nedeniyle yönlendirilen bu ışın ve geçmişteki faz kaymaları RF frekansında sık sık uygulanıldı.

Böylece, kapasitenin en uygun şekilde kullanılması ve de aynı zamanda da engellenmesi sağlanacaktır.

Modern ışın şekillendirme, dizi antenleri örüntüyü belirli bir optimum (en uygun) kriter boyunca şekillendirir. Bu sistemler, bir başka deyişle akıllı anten olarak adlandırılır. Akıllı terimi, belirli durumlara göre ışın örüntüsünün yönlendirilmesi anlamını ifade eder. Akıllı, daha genel olarak anten performansı hesaplama kontrolü anlamına da gelir. Burada, antenlerin akıllı olması söz konusu değildir. Sistem, genel olarak düşünüldüğünde algoritmaların işaret işlemciler kullanılarak gerçeklenmesi sistemi akıllı hale getirecektir [14].

Akıllı antenlerin, geliştirilmiş radar sistemleri ve gezgin haberleşme ile birlikte sistem kapasitesini geliştirmek için Uzay Bölüşümlü Çoklu Erişim (SDMA) yöntemi ile kullanımı önerilir. Aynı zamanda akıllı antenlerin, mevcut Frekans Bölüşümlü Çoklu Erişim (Frequency Division Multiple Access, FDMA), Zaman Bölüşümlü Çoklu Erişim (Time Division Multiple Access, TDMA), Kod Bölüşümlü Çoklu Erişim (Cod Division Multiple Access, CDMA) gibi çoklu erişim yöntemlerine olan uyumluluğu söz konusudur [15].

Akıllı anten örüntüyü belirli kriter temelinde algoritmalar yoluyla kontrol eder. İlgili işaret boyunca örüntüyü oluşturur. Girişim işaretleri için boş örüntü oluşturur. Ve de istenen işaretin daha sağlıklı bir şekilde alınması sağlanır. Genel olarak sayısal işaret işleme kullanılarak algoritmaların gerçek zamanlı çalışması başarılır.

Dizi çıkışları analog sayısal dönüştürücüleri (A/D) kullanılarak sayısallaştırılır. Bu sayısallaştırma işlemi IF temel bandında gerçekleştirilir. Işın örüntüsü, sayısal işaret işleme ile gerçekleştirildiği için bu işlem sık sık sayısal işaret işleme olarak belirtilir.

2.3 Akıllı Anten Sistemi ile Geleneksel Anten Mimarisi Karşılaştırılması

Geleneksel anten mimarisi günümüzde hala kullanımı devam etmekte olan mimaridir. Bu mimaride anten çok büyük doğrultularda yayılım yapmaktadır. Örneğin 120°' lik makro sektörler kullanan günümüzdeki antenlerde kullanıcı antenden çok az

miktarda enerji alabilmektedir ve de kullanıcı doğrultusu haricindeki örüntü boşa harcanmaktadır.

Aynı zamanda kullanıcı haricindeki örüntü anten tarafından istenmeyen girişimlerin alınması anlamına gelmektedir. Bu hem enerjinin boşa harcanması hem de girişim gibi problemlerin ortaya çıkması anlamına gelmektedir.

Kanal faktörü de haberleşme sistemlerinde önemli bir etkiye sahiptir. Ortamın durumu bu yüzden geleneksel mimarinin geliştirilmesi sonuçlarını beraberinde getirmiştir. Çünkü haberleşme ortamı önceden tahmin edilebilecek özelliklere sahip değildir. Gezgin istasyonun sürekli hareket halinde olması ve de farklı haberleşme ortamı özelliklerinden haberleşme olumsuz yönde etkilenecektir. Bu yüzden mevcut geleneksel mimari bu problemlerin çözümü için yeterli değildir.

Akıllı anten sistemleri yukarıda açıkladığımız sebeplerden dolayı geliştirilmiştir. Özellikle ortamda bulunan gürültü, girişim ve çokluyolllardan dolayı akıllı anten mimarisi önemli ölçüde faydalı olacaktır. Yalnızca istenen işaret doğrultusunda örüntü oluşturularak istenmeyen gürültü, girişim ve de çokluyol etkisi azaltılabileceği gibi istenmeyen işaret doğrultusunda boş örüntü de oluşturularak sistem daha avantajlı kullanılabilir.

Kanal durumu göz önüne alındığında, akıllı anten sistemlerinin geleneksel antenlere göre faydaları şu şekilde özetlenebilir.

Öncelikle sistem farklı çoklu erişim yöntemlerine uyumluluğu ile ön plana çıkar. FDMA, TDMA, CDMA gibi sitemler ile SDMA gibi gelecek nesil erişim yöntemine uygun olarak tasarlanmıştır. Özellikle SDMA yöntemi uzaysal boyut anlamına gelip aynı fiziksel alanda eş zamanlı olarak birden fazla kullanıcının kullanımı söz konusudur. SDMA sistemlerin en önemli avantajı gözle görülür ölçüde kapasite artışı oluşturmasıdır.

Maliyet açısından ele aldığımızda geleneksel antenlere göre daha iyi sonuçlar ortaya koymaktadır. Başlangıçta karmaşık yapısı itibariyle geleneksel antenlere göre daha

maliyetli görünse de olumlu getirisi en önemli tercih sebebini oluşturmaktadır. İçerisinde bulunan işaret işleme algoritmaları için gerekli olan DSP ve FPGA gibi işaret işleme elemanlarının günümüzde çok ucuza üretilmesi akıllı anten mimarisinin maliyetinin daha az olması anlamına gelmektedir.

Akıllı anten algoritmalarının sonuçları göz önüne alındığında enerjinin uygun şekilde kullanımından akıllı anten sistemlerinin seçiminin çok daha uygun olduğu kolaylıkla fark edilebilir. Çünkü uygun ışıma örüntüsünün oluşturulması ile enerjinin tasarruflu kullanımı söz konusu olacaktır.

Akıllı anten sistemleri işaret-gürültü oranı (signal to noise, SNR) ve de işaret-girişim oranı (signal to interference, SIR) için iyi bir performans gösterir. Örüntü istenen kullanıcı yönünde olduğundan işaret seviyesi artarken, girişim işareti doğrultusunda boş örüntü oluşturulduğu için girişim seviyesi düşmektedir. Böylece yukarıda belirttiğimiz gibi işaret girişim seviyesi önemli ölçüde yükselmektedir [16].

2.4 Akıllı Anten Sistemleri Yapısı

Öncelikle istenen işaretin yakalanması ve istenmeyen girişimlerin engellenmesi için haberleşme algoritmalarının iyi bir şekilde çalışmasını sağlayacak iyi bir işaret işleme mekanizması gereklidir. Bu işaret işleme kısmı akıllı antenlerin temel yapsını oluşturmaktadır. Yüksek performansa sahip farklı işlemciler ve gelişmiş akıllı anten algoritmaları kullanımı ile bu performas maksimum seviyeye çıkarılabilir.

İlk olarak, bu mekanizma akıllı anten sistemlerini diğer geleneksel anten mimarilerinden ayıracak önemli bir nokta olacaktır. Bu kısımda yapılan çalışmada hızından dolayı FPGA tercih ettik. FPGA' ler, diğer mikroişlemci ve DSP' lere göre hız açısından daha iyi sonuçlar ortaya koymaktadır.

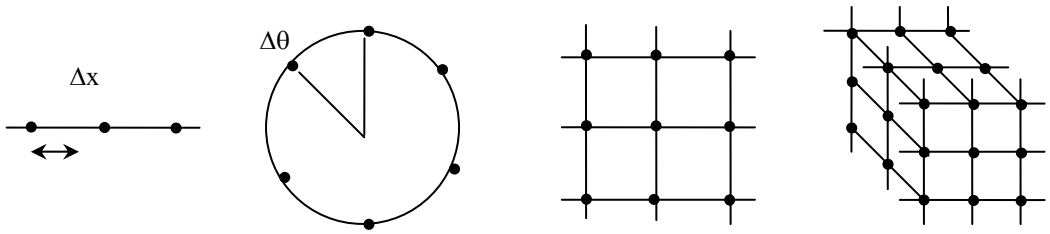
Burada önemli bir özellik ise mevcut işaret işleme yeteneğine sahip DSP ve FPGA gibi işaret işlemcilerin çalışabilmesi için antenlerden alınan işaretlerin sayısal işaretlere dönüştürülmesi gerekliliğidir. Çünkü işaretler antenlerden yüksek frekans değerleri ile birlikte alınır, analog işaret alırlar, fakat isminden de anlaşılacağı üzere

sayısal işaret işlemcileri belirli örnekleme frekansına göre sayısal verinin işlenmesi işlemini gerçekleştirirler. Şimdi maddeler halinde akıllı anteni oluşturan bölümler incelenecek.

2.4.1 Anten dizileri

Anten dizileri örüntünün oluşturulması için en önemli yapıya sahiptirler. Örüntünün istenen kullanıcı doğrultusunda oluşturulması için işaret işleme kısmından gelen ağırlık vektörleri (katsayıları) ile güncellenen bilgi sayesinde istenen kullanıcı doğrultusunda örüntü oluşturulur [17].
Anten dizilerinin fiziki olarak hareketliliği söz konusu değildir. Anten dizileri ağırlık katsayıları ile örüntünün fazında ve genliğinde değişiklilik oluşturur. Bu değişiklik akıllı anten algoritmalarının kullanımıyla gerçekleştirilir.

Şekil 2.1'de ortamın durumunda diziler düzgün Doğrusal Anten Dizisi (Uniform Lineer Antenna Array, ULA), Düzgün Dairesel Anten Dizisi (Uniform Circle Antenna Array, UCA), İki Boyutlu Anten Dizisi (Two Dimensional Antenna Array) ve Üç Boyutlu Anten Dizisi(Three Dimensional Antenna Array) olarak adlandırılır.



Şekil 2.1: Akıllı anten dizileri

2.4.2 Radyo frekans katı

Haberleşmenin gerçekleştirildiği ortamda birçok işaret bulunabilir. Bu sebeple akıllı anten sistemleri diğer kullanıcılardan gelen girişim işaretlerini de alabilir. Bu girişim işaretleri azaltıldıktan sonra işaret işleme işleminin gerçekleştirilmesi gerekir. Bu yüzden ışın şekillendirme algoritmaları kullanarak işaret işleme işlemi

gerçekleştirilmeden önce RF katında bulunan işaretin gürültü ve de istenmeyen girişimlerden temizlenmesi işlemi radyo ferkans katı kullanılarak gerçekleştirilir.

Burada, asıl amaç gürültü ve girişimleri en aza indirgeyerek alıcı duyarlılığını arttırmaktır. Bu yüzden, burada Düşük Gürültü Yükselteci (Low Noise Amplifier, LNA) kullanılır. LNA' nın hem gürültü hem de girişimlere karşı daha iyi bir başarım sağlaması için, düşük gürültü sayısına ve yüksek kazanca sahip olması gerekir [18].

2.4.3 Aşağı dönüştürücü

Akıllı anten algoritmalarının DSP, FPGA gibi sayısal işaret işleyiciler kullanılarak gerçekleşmesi için işaretin frekansının belirli değerlere indirilmesi gerekir. Çünkü kullanılan sayısal işaret işlemcileri sınırlı örnekleme frekansı kullanarak algoritmaları gerçekleyebilir. Akıllı anten gibi karmaşık sistemlerde aşağı dönüştürme işlemi yüksek başarıma sahip karıştırıcılar tarafından gerçekleştirilir. Intermodülasyon bozulması gibi akıllı antenlerin performansını olumsuz yönde etkileyen bozulmaların önüne geçebilmek için LNA ile karıştırıcı arasına zayıflatıcı konulması gerekir. Böylece, RF giriş gücünün, intermodülasyon bozulmaya neden olmayacak şekilde kalması sağlanır [19].

2.4.4 Sayısal işaret işleme modülü

Sayısal işaret işleme modülü akıllı anten sistemlerinin en önemli kısmını oluşturur. Hatta şu da söylenebilir ki akılllı antenin isminin ne ifade ettiği bu bölümden kolaylıkla anlaşılabilir.

Öncelikle, sayısal işaret işleyici modül örüntünün istenen kullanıcı yönünde oluşturulmasını sağlayarak girişimlerin engellenmesini sağlayan yüksek performansa sahip algoritmaların çalıştığı kısımdır.

Gezgin haberleşme sistemleri sürekli hareket halinde bulunabilme kabiliyetine sahip oldukları için ışın örüntüsünün istenen kullanıcıyı hızlı bir şekilde takip etmesi ve de girişimleri de aynı hızda engellemesi gerekir. Bu sebeple gerçek zamanlı işlemlerin

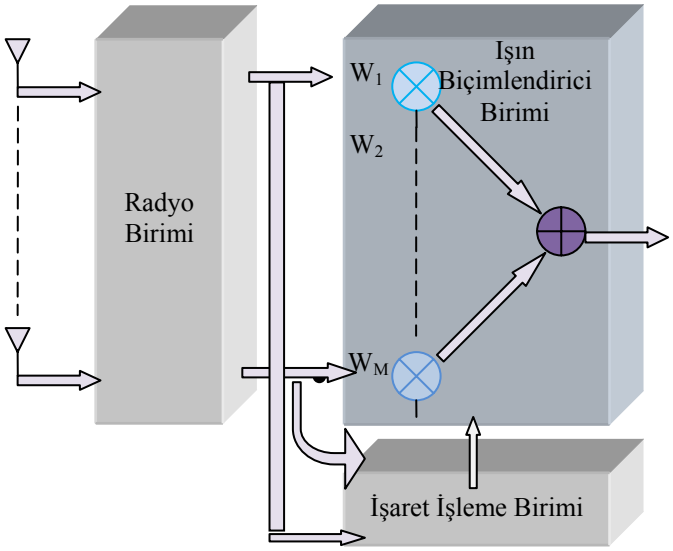
gerçeklenebilmesi için yüksek hız performansına sahip DSP ve FPGA gibi işaret işlemcilerin kullanılması gerekir. Özellikle yapılan çalışmada DSP' ye göre çok daha iyi hız performansına sahip olan FPGA kullanıldı.

FPGA' in lojik bir donanıma sahip olması ve algoritmaların karmaşıklığı yapılan sistemin daha zor gerçeklenmesi anlamına gelmektedir. Genel olarak sistemin performasını en iyi şekilde karşılayacak algoritmalar bu birimde tercih edilir. FPGA tercih edilişi zaman açısından DSP' lere göre çok daha büyük hız performansı sağlar. Fakat daha önce de belirtildiği gibi bu mimaride algoritmaların tasarlanabilmesi için basit matematiksel ifadeler kullanılmalıdır.

Yapılan çalışmada bir sonraki bölümde de geniş bir şekilde anlatılacak En Küçük Karesel Ortalama (Least Mean Square, LMS), Sabit Modül(Constant Modulus, CM) ve de Uzaysal Kod Korelatör (Space Kod Korelatör, SCC) algoritmaları kullanılmıştır. Bu algoritmalardan LMS ve CM iteratif algoritma özelliklerine sahip olup, SCC algoritması ise çok fazla matematiksel matris işlem yüküne sahiptir. Özellikle matematiksel işlem yükünün fazla olması zaman açısından kayıplara neden olabilir. Bu sebeple de yüksek hız performansı ile daha geniş kapasiteye sahip Xilinx ailesine ait Virtex II ve Virtex4 işlemci kullanılmıştır. Bu işlemcilerden VirtexII yine de SCC' deki matematiksel işlem yükünü karşılayacak kapasiteye sahip değildir. Bu sebeple bir noktadan sonra yetersiz kapasiteden dolayı Virtex4 kullanilarak işlemler paralel mimari ile gerçekleştirilmiştir.

2.5 Akıllı anten alıcı verici yapısı

Yukarıda akıllı anten yapısı verildikten sonra bu vermiş olduğumuz bilgiler doğrultusunda akıllı anten alıcı ve verici yapısını gözden geçirelim. Öncelikle bu iki mimari biribirine çok benzemektedir. Bu mimarilerden alıcı anten mimarisine değinelim. Öncelikle anten dizi kısmı bizim algoritmamızda belirlediğimiz 5 anten içermektedir. Böylelikle 5 anten elemanından gelen işaret akıllı anten çıkışında da gözükeceği üzere 1 elemana dönüştürülmektedir. Akıllı anten alıcısı şekilde de belirtildiği üzere radyo birimi, ışınbiçimlendirici ve işaret işleme birimlerinden oluşur. Şekil 2.2' de anten alıcısı gösterilmiştir.



Şekil 2.2: Akıllı anten alıcısı

Radyo birimi aşağı dönüştürücü ve Analog-Sayısal Dönüştürücü (Analog Digital converter, ADC) içermektedir. Daha önceki kısımlarda da anlatıldığı üzere, aşağı dönüştürücü kısmı karıştırıcılar ile gerçekleştirilir. ADC işaret işleme mekanizmasının sayısal işaret işleme gerekliliğinden dolayı, önemli bir fonksiyona sahiptir. Özellikle, radyo birimi analog olarak alınan işaretin işaret işleme birimiminde işlenebilecek özelliklere sahip olmasını sağlar.

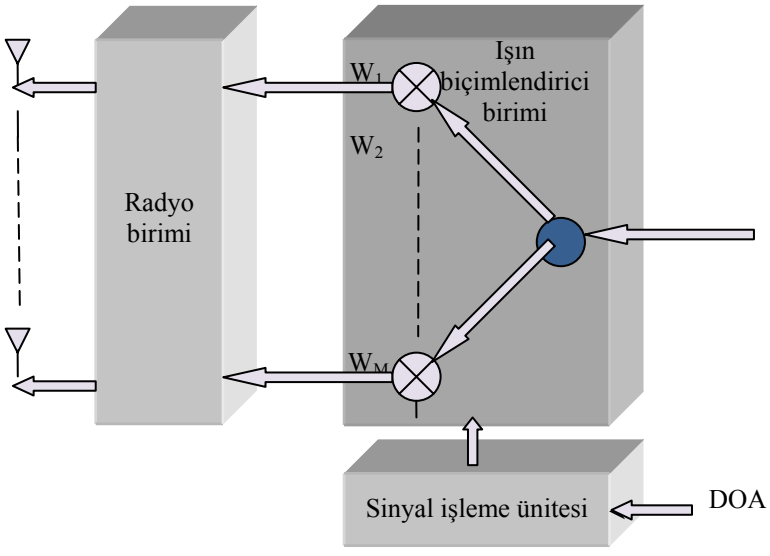
İşaret işleme birimi esas işlevselliğin olduğu kısımdır. Özellikle ışın biçimlendirme algoritmalarının FPGA ve DSP gibi yüksek performansa sahip işlemciler ile işlenmesi bu bölümde gerçekleştirilir. Alınan işarete bağlı olarak anten elemanlarının herbirinden alınan işaret ile çarpılan karmaşık ağırlık vektörleri hesaplanır. Bu antenler yukarı bağlantı yönünde anten örüntüsünü belirler.

Özellikle, ışın biçimlendirme bölümü bizim de çalışmamızın en önemli kısmını oluşturmaktadır. İstenen kullanıcı yönünde örüntünün oluşması için ağırlık vektörlerinin dikkatli bir şekilde belirlenmesi gerekir. Basit bir algoritma örneği vererek bu işlemin daha iyi anlaşılmasını sağlayabiliriz. FPGA de uyguladığımız En Küçük Kareler Yöntemi (Least Mean Square, LMS) ve Sabit Modül (Constant Modulus, CM) algoritmalarını inceleyelim. Bu algoritmaların çıkışları belirli kriter değer ile karşılaştırılmalıdır. Eğer algoritmalar ile elde ettiğimiz çıkış değeri kriter

değerden küçük ise algoritma sonlandırılır. Eğer tam tersi söz konusu ise, istenen kullanıcı yönünde ışın örüntüsünün oluşturulabilmesi için algoritmanın çıkışı kriter değerden küçük olana kadar algoritma devam ettirilir.

Şekil 2.3' de anten alıcısı gösterilmiştir. Akıllı anten vericisi daha önce de ifade ettiğimiz gibi alıcı yapısına çok benzemektedir. En önemli fark Radyo Biriminde Analog Sayısal Dönüştürücü (Analog Digital Converter, ADC) ve aşağı dönüştürme birimleri yerine Sayısal Analog Dönüştürücü (Digital Analog Converter, DAC) ve yukarı dönüştürücü kullanılmasıdır. İşaret, ışın biçimlendirme ünitesinde karmaşık ağırlık vektörleri ile ağırlandırılarak anten eleman sayısı kadar kanala verilir. Burada ADC ve DAC' lar, işlem yapılacak işaret işlemcilerin özelliklerine göre Analog-Sayısal ve Sayısal Analog dönüşüm işlemlerini belirli örnekleme frekansı kullanarak gerçekleştirir [20].

Çalışmada gerçek zamanlı olarak antenden elde edildiği varsayılan analog işaretler FPGA' de işlem yapabilmek için sayısal işaretlere dönüştürüldü. Burada standart olarak IEEE tarafından belirlenen IEEE 754 kayan noktalı sayı formatı kullanıldı. Burada gerçekte yapılan işlem ADC' den elde edilen işaretlerin İşaret İşleme Biriminde uygun algoritmalarla işlenmesi gibi düşünülebilir. Bu sebeple de ADC ve DAC' larla bu çalışmada ilgilenilmedi.



Şekil 2.3: Akıllı anten vericisi

Çalışma prensibi olarak karşılaştırıldığında en önemli fark ise aşağı bağlantıda kanal ile ilgili bilgiye sahip olunmamasıdır. Yukarı bağlatıda kanal ile ilgili bilgiye sahibiz. Fakat aşağı bağlantıda kanal hakkında bilgiye sahip olmamamız, kanalda bulunan gürültü ve girişimlerden sistemimizin olumsuz etkilenmesi anlamına gelecektir. Yapılan çalışmada işlemlerimizin yukarı bağlantıda olduğunu düşünerek algoritmalarımızı belirledik.

Zaman Bölüşümlü Çoğullama sistemlerinde (Time Division Dublex, TDD) gezgin istasyon ile temel istasyon farklı zaman dilimlerinde aynı taşıyıcı frekansı kullanırlar. Bu sebeple yukarı bağlantı ile aşağı bağlantı arasında hesaplanan ağırlık vektörlerinin kanalda bir değişim olmassa aynı olması söz konusudur. Fakat bu çok düşük bir olasılıktır. Çünkü temel istasyon sabit olmasına rağmen (gezgin kullanıcının hareketi ile başka bir temel istasyon aktif olabilir) gezgin istasyon sürekli seyir halinde olduğundan haberleşme ortmanın sürekli değişmesi söz konusu olacaktır. Bu sebeple de yukarıda ifade edildiği üzere aynı hesaplama değerleri çoğu zaman kullanılamayacaktır.

2.6 Akıllı Anten Çeşitleri

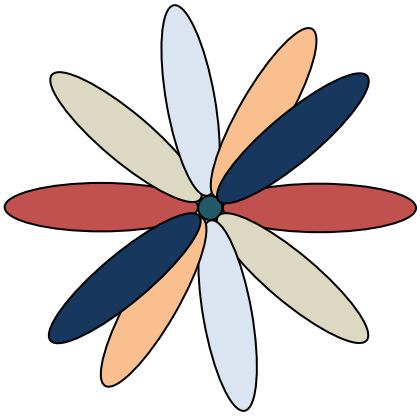
Akıllı anten sistemlerinin çeşitleri şu şekilde sıralanabilir.

- Faz Dizileri Sistemleri (Phase Arrays Systems)
- Uzaysal İşleme Sistemleri (Spatial Processing Systems)
- Sayısal Işın Biçimlendirme Sistemleri(Digital Beamforming Systems)
- Uyarlanabilir Anten Sistemleri sistemleri (Adaptive Antenna Systems)
- Anahtarlamalı Anten Sistemleri(Switched Antenna Systems)

Temel yapı olarak birbirilerine çok benzeyen bu sistemler çok küçük farklar ile biribirinden ayrılır. Bu anten sistemlerinden kullanımı yaygın olan ve de literatürde sık karşılaşılan Anahtarlamalı Anten Sistemleri (Switched Antennna Systems) ile Uyarlanabilir Anten Sistemleri (Adaptive Antennas Systems) üzerinde durulacak.

2.6.1 Anahtarlamalı anten sistemleri

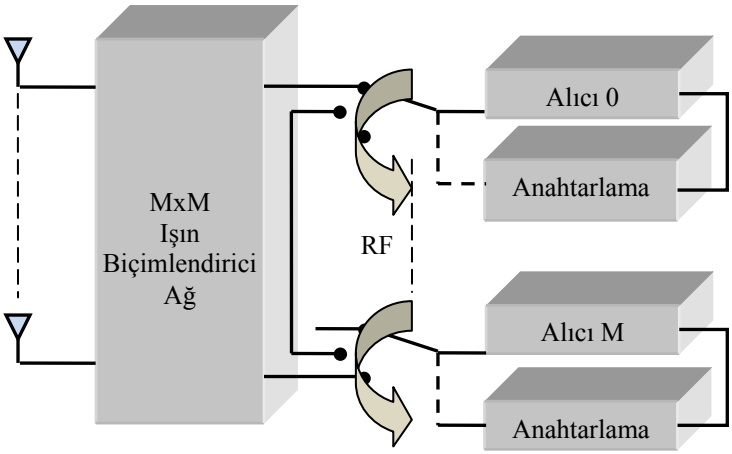
Anahtarlamalı anten sistemleri günümüzde kullanılan hücresel senkronizasyona sahip sistemlere benzemektedir. Makro sektörlere bölünmüş geleneksel mimariye sahip antenlerin mikro sektörlere bölünmesi gibi düşünülebilir.



Şekil 2.4: Mikrosektörlere ayrılmış anten

Geleneksel anten mimarileri makro sektörlere bölünmüş olup, bu makrosektörler 120°' lik açılarla bulunur. Fakat günümüzde haberleşme ihtiyacının artması ile makro sektörlere bölünmüş antenler gereksinimleri karşılayamaz duruma gelmiştir. Bu sebeple ilk aşamada makro sektörlerin mikro sektörlere bölünmesi düşünülmüştür. Yapısı oldukça basit olup, önceden belirlenen ışın örüntülerine göre istenen işaretin anahtarlanması anlamına gelir. Geleneksel anten mimrilerine göre çok daha fazla sektöre bölündüğünden dolayı sistem performansı karşılaştırıldığında çok daha iyi sonuçlar verir [21].

Çeşitli basit algoritmalar kullanılarak çok kolay bir şekilde anahatarlama işlemi gerçekleştirilebilir. Şekil 2.4' de mikro sektörlere ayrılmış anten mimarisi gösterilmiştir.



Şekil 2.5: Anahatarlamlı anten sistemi

Çok düşük maliyet gerektirmesi tercih edilmesinin en önemli sebeplerindendir. Yapı yalnızca daha önce belirlenen ışın örüntülerine istenen işaret doğrultusunda anahtarlama olduğu için çok karmaşık bir sisteme gerek duyulmadan basit algoritmalar ile kolaylıkla bu işlem gerçekleştirilebilir. Bu yüzden hem gerçekleştirilen algoritmalar (yalnızca istenen işarete anahatarlama özelliğine sahip olması yeterli) hem de kullanılan devre elemanları açısından (anahtarlar özellikle) diğer sistemlere göre tercih edilirdi. Şekil 2.5' de anahtarlamalı anten sistemi gösterilmiştir.

Bu sistem, ışın örüntüleri daha önceden belirlendiği için gezgin haberleşme hareketinden istenmeyen durumlar ortaya çıkabilmektedir. Gezgin haberleşme hareketinden dolayı anahtarlama işlemi sağlıklı bir şekilde gerçekleştirilemeyecektir. Bunun yanında mikrosektörler önceden belirlendiğinden dolayı istenen işaret örüntüsü tam olarak yakalanamayabilir, kanalda bulunan girişim yakalanılarak istenen işaret gibi işlem (anahtarlama) yapılır. Bu da sistemin hatalı sonuçlar vermesini aynı zamanda bu istenmeyen durumların artması sonucu haberleşmenin yapılamaması anlamına gelmektedir. Özellikle ortamda çok fazla sayıda girişim bulunmasından ve de engebe ve fiziki şartlarlardan dolayı oluşan çokluyol etkilerini dayanıksızlığından yerini uyarlamalı anten sistemleri (adaptive antenna systems) bırakmıştır.

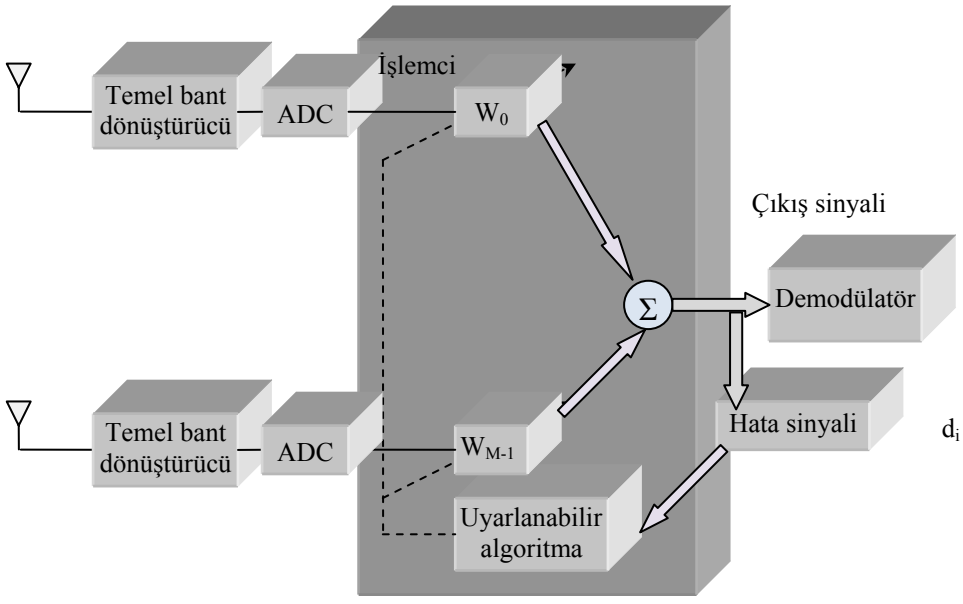
2.6.2 Uyarlanabilir anten sistemleri

Uyarlanabilir anten sistemleri dinamik özelliklere sahip olduğundan, ışın örüntüsünün gelişmiş işaret işleme teknikleri kullanılarak istenen işaret doğrultusunda oluşmasını sağlar.

Uyarlanabilir anten sistemlerinde ışın örüntüsü sürekli istenen işareti yakalamak amacıyla güncellenir. Bu da özellikle sitemin girişimlere karşı iyi bir performas sağlaması anlamına gelmektedir.

Yeni nesil haberleşme sistemlerinde uyarlanabilir anten sistemlerinin kullanımı ile istenen işaretin izlenebilmesi istenmeyen girişimlerin de boş örüntü oluşturularak engellenmesi aynı zamanda sistemin kapasitesini arttıracak ve de kanalın çok fazla sayıda kullanıcı tarafından kullanılması anlamına gelecektir.

Ayrıca gezgin istasyon sürekli seyir halinde bulunabileceğinden dolayı adaptif olarak ışın örüntüsünü değiştirebilen uyarlanabilir anten sistemleri işaretin minimum hata ile yakalanabilmesini sağlayabilir. Şekil 2.6'da uyarlamalı anten sistemi çizilmiştir.



Şekil 2.6: Uyarlanabilir anten sistemi

Öncelikle işaretin işlenebilmesi için belirli frekans değerlerine indirilmesi gerekir. Bu sebeple antenden alınan işaret için temel band seviyesine dönüştürme işlemi gerçekleştirilir. Daha sonraki kısım ise işaretin işlenebilmesi için sayısal verilere dönüştürülmesidir. Bu kısım analog sayısal dönüştürücüler sayesinde gerçekleştirilir. Esas kısımda ise işaret işleme algoritmaları çalıştırılır. Ve de ağırlık bilgileri bulunarak güncelleme işlemi gerçekleştirilir [22].

Uyarlanabilir anten sistemleri ile anahtarlamalı anten sistemleri karşılaştırıldığında uyarlanabilir anten sistemlerinin çok daha iyi özelliklere sahip olduğu görülür. Ayrıca performans açısından karşılaştırıldığında da istenen işaretin daha doğru tahmin edilebilmesi için uyarlanabilir anten sistemlerinin tercih edilmesi gerekir.

Uyarlanabilir anten sistemlerinin dinamik olarak istenen işareti takip etmesi işaret gürültü (Signal to Noise, SNR) oranınını iyileştirmesi ve de ortamda bulunan girişimlerin üstesinden gelebilmek için işaret girişim (signal to interfrence, SIR) oranınını maksimum yapabilmesi nedeniyle önemli avanatajlara sahiptir. Çünkü uygun ağırlık vektörleri yazılım ile güncellenebildiğinden istenen işaret doğrultusunda örüntünün oluşturulması, istenmeyen girişimler doğrultusunda da boş örüntülerin oluşturulması istenen işaretin yakalanması ve de girişimlerin engellenmesi için çok önemlidir.

İşlevsel olarak anahtarlamalı anten sistemlerine göre karşılaştırıldığında ise, anahtarlamalı anten sistemleri yalnızca istenen işaretin faz bilgisine göre işlem yaparken, uyarlanabilir anten sistemleri ise hem faz hem de genliğinde değişiklikler sağlayarak işaretin yakalanmasını sağlar.

Uyarlanabilir anten sistemleri dinamik özelliğinden ortamdaki değişimlerden en az etkilenir yukarıda da belirtildiği üzere çoklu yol etkilerine karşı çok daha dayanıklı bir durum sağlar. Fakat anahtarlamalı anten sistemleri için ışın örüntüsü daha önceden belirlendiğinden dolayı çokluyollar sistemi çok fazla etkiler ve de haberleşmenin zaman zaman yapılamaması anlamına gelir.

Yapılan çalışmada dinamik olarak istenen işaret doğrultusunda izleme yapabileceğimiz uyarlamalı anten sistemleri kullanıldı. Anahtarlamalı anten sistemleri yalnızca anahtarlama işlemi yapabilme özelliğine sahip olup, yapılan işlem yalnızca daha önceden belirlenen mikro sektörlere göre işaretin anahtarlanmasıdır. Bu yüzden daha gelişmiş işaret işleme birimine sahip, gelecekte de kullanımı yaygınlaşacak uyarlamalı anten sistemleri üzerinde LMS, CM ve de SCC algoritmalarını kullanıldı.

Özellikle yapılan çalışmada yalnızca işaret işleme kısmı üzerine yoğunlaşılmış olup, antenden alınan işaret temel band ve de sayısala dönüştürülmüş kabul edilir. Bundan sonraki aşamada ise, FPGA kullanılarak gerçekleme yapılmıştır.

BÖLÜM 3. ALAN PROGRAMLAMALI KAPI DİZİLERİ

3.1 Giriş

Alan Programlamalı Kapı Dizileri (FPGA) günümüzde yaygın kullanım alanına sahiptir. Yüksek hız performansı nedeniyle bir çok alanda diğer mikroişlemci ve Sayısal İşaret İşlemci'lere (Digital Signal Processing, DSP) göre çok daha fazla tercih edilir hale gelmişlerdir. Özellikle paralel çalışma mimarisinin mevcut olması hızın artmasının en önemli sebeplerinden biri olarak karşımıza çıkmaktadır. En temel mimarilerden biri olan Uygulamaya Özel Tümleşik Devre (Application Specific Integrated Circuit) ile karşılaştırıldığında FPGA' lerin yazılım esnekliğine sahip olması fakat ASIC' lerin yalnızca hazırlanan sistem için fonksiyonelliğe sahip olması FPGA' lerin tercih ediliş sebeplerini açıkça ortaya koymaktadır. Yine DSP' ler ile karşılaştırıldığında çok daha iyi bir hız performansı sağlaması paralel mimari kulanılma özelliği ile açıklanabilir [23].

Bu bölümde ilk olarak genel programlanabilir çiplerin gerekliliğinden bahsedilecek. Öncelikle yapılan çalışmada kullanılan çok hızlı işlem yapabilme kabiliyetine sahip FPGA' ler hakkında kısa bilgi verdikten sonra FPGA' leri programlamak için kullanilan Yüksek Hızlı Tanımlama Dili (VHDL) tanıtılacak . Bu bölümde yazılan koddan örnekler verilecek. Son kısımda ise, kullanılan Xilinx ailesine ait XilinxII hakkında katalog bilgisi sunulacaktır.

3.2 Programlanabilir Çipler

Günümüzde teknolojinin hızla gelişmesiyle birlikte insanoğlunun da istekleri gün geçtikçe artmaktadır. Özellikle tasarlanan sistemlerin hızlı bir şekilde çalışması ve doğru sonuçların alınması gerekmektedir. Aynı zamanda sistem tasarımı optimum tasarım kriterlerine göre gerçekleştirilmek zorundadır. Örnek vermek gerekirse tasarım hem hızlı hem doğru sonuçlar verecek şekilde hem de minimum maliyetle yapılmalıdır. Aynı zamanda az yer kaplaması gibi faktörleri ekleyerek istekler

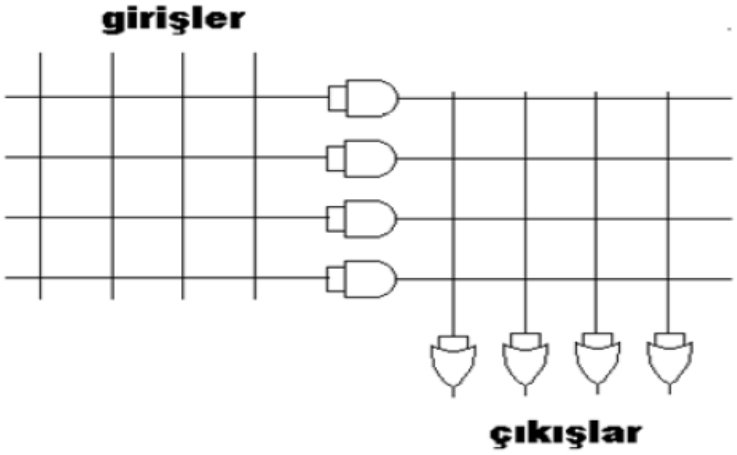
sonsuza çıkarabilir. Bu nedenle yukarıda sayılan özellikler insanları tümleşik devrelere (çip) yönlendirmiştir. Özellikle günümüzde uygulama alanları çok geniş olan (yalnızca uygulandığı sisteme özel) ASIC' ler geliştirilmiştir. ASIC' ler hem üretim maliyeti hem de hızı sebebiyle çok tercih edilen mimarilerdir. Seri üretim ile çok kolay ve de hızlı bir şekilde üretilmeleri ASIC' lerin kullanılmasının diğer avantajları arasındadır. Fakat adından da anlaşılacağı üzere ASIC (Application Specific Integrated Circuit) yalnızca tasarlandığı devreye özel kullanabileceğinden dolayı tasarım değiştirilme ihtiyacı duyulduğu zaman tekrar programlanabilme yeteneğine sahip değildir. Bu ASIC' lerin en önemli dezavantajlarından birini oluşturmaktadır. Bu sebeple programlabilen devre elemanlarına ihtiyac duyulmuştur. Çünkü programlanabilir devre elemanları yalnızca tasarıma özgün değil, aynı zamanda programlama yetenekleri ile geniş bir kullanım alanında defalarca kullanılabilir. Genel olarak programlama yeteneğine sahip devre elemanları haberleşme, kontrol gibi farklı alanlarda malzeme ömürleri ile sınırlı olarak işlevsellik kazanabilir. Bu mimariler Basit Programlanabilen Mantık Devreleri (Simple Programmable Logic Devices, SPLD), Karmaşık Programlanabilir Mantık Devreleri (Complex Programmable Logic Devices, CPLD), Alan Programlamalı Kapı Dizileri (Field Programmable Gate Array, FPGA) ve de Alan Programlamalı İç Bağlantı (Field Programmable Interconnect, FPIC) olarak adlandırabilir [24]. Bu yapılar hakkında kısaca bilgi verilecek.

3.3 Basit Programlamalı Mantık Devreleri (Simple Programmable Logic Devices)

İsminden de anlaşıldığı üzere gibi programlanabilir devrelerin en basit fonksiyona sahip olanıdır. Basit özelliklere sahip olmasının yanında maliyeti de esas tercih edilme sebeplerinden biridir. Fakat basit işleve sahip devrelerde oldukça çok kullanılmaktadır.

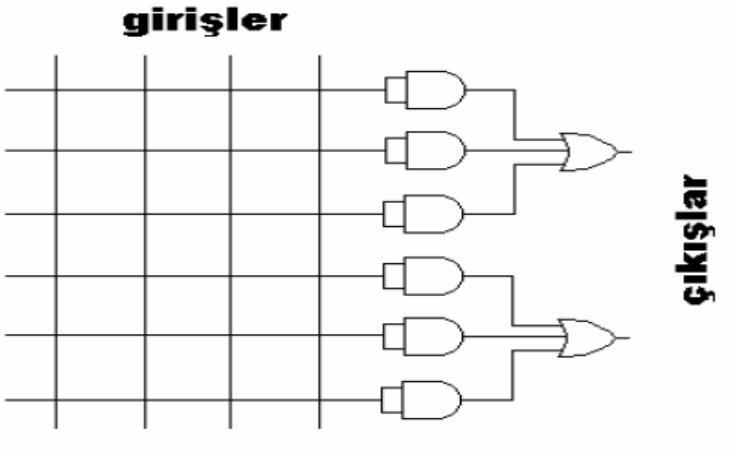
Genel olarak düşünüldüğünde bu devreler iki alt bölüme ayrılabilir. Programlamalı Mantık Dizileri (Programmable Logic Arrays, PLA) ve Programlamalı Mantık Devreler (Programmable Logic Device, PLD) olarak ayrılabilir [24].

Şekil 3.1' de gösterilen PLA' ler herhangi bir AND veya OR kombinasyonu ile gerçekleştirilebilen iki adet programlanabilme düzenine sahiptir. Burada önemli bir diğer özellik ise, farklı OR kapıları kullanılarak AND işleminin gerçeklenebilme özelliğidir.



Şekil 3.1: PLA

Şekil 3.2' de gösterilen PAL ise programlanabilme özelliğine sahip olup, belirli sayıda AND OR kapıları kombinasyonları ile oluşturulur. Bu devreler PLA lara göre oldukça yavaştırlar.



Şekil 3.2: PAL

3.4 Karmaşık Programlamalı Mantık Devreleri (Complex Programmable Logic Devices)

Basit Programlamalı Mantık Devrelerinin yetersizliğinden Karmaşık Programlamalı Mantık Devreleri ortaya çıkmıştır. İyi incelendiğinde mantık olarak aynı yapıya sahip olmalarına rağmen içerisinde çok fazla sayıda Basit Programlamalı Mantık

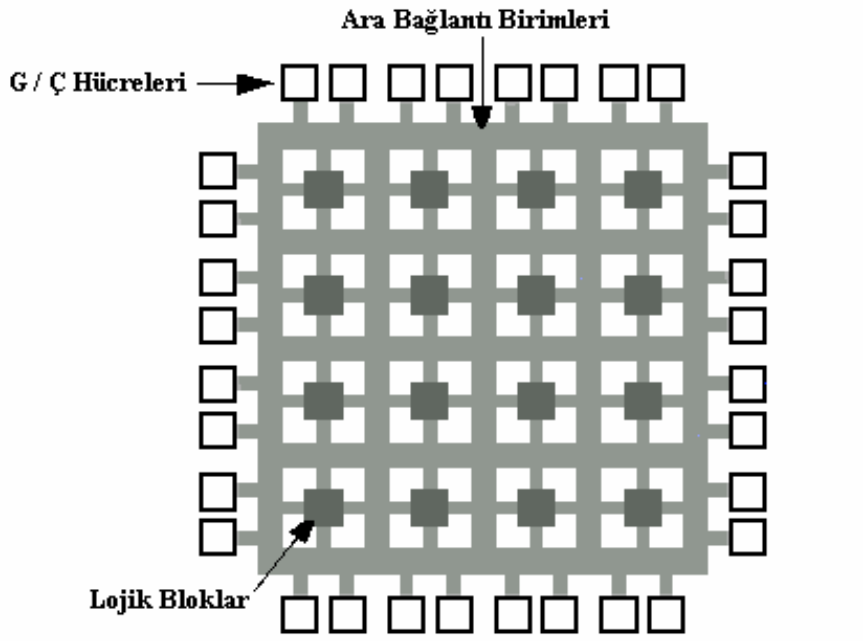
Devrelerinin bulunduğu düşünülebilir. İçerisinde 200 kapı bulunabileceğinden dolayı çeşitli kombinasyonlar ile hızlı işlemler yapılabilir. Böylece sistemin esnekliği artacak daha karmaşık işlemler yapılabilir olacaktır [24].

3.5 Alan Programlamalı Kapı Dizileri (Field Programmable Gate Array)

Günümüzde hızından dolayı yaygın olarak kullanabilen geniş uygulama alanına sahip programlanabilir devre elemanlarıdır. Paralel işlem yapabilme kabiliyetine sahip olduklarından dolayı diğer işlemcilere göre çok daha iyi hız performansına sahiptirler.

İçerisinde çok fazla sayıda kapı bulunabildiğinden dolayı esnek işlem yapabilme kabiliyetine sahiptirler. Tabi ki kapılardan oluştuğu için yalnızca sayısal işlem yapabilme kabiliyeti zaman zaman çeşitli zorluklarla karşılaşılması anlamına gelir. Özellikle yazılan program çok karmaşık matematiksel işlemlerden oluştuğu için kapı gecikmelerinden dolayı birçok problem ile karşılaşılaşıldı.

FPGA' ler Düzenlenebilir Mantık Blokları (Configurable Logic Block, CLB), Giriş Çıkış Blokları (Input Output Block) ve İç Bağlantı Blokları (İn Conncetion Block ICB) elemanlarından oluşur [25].

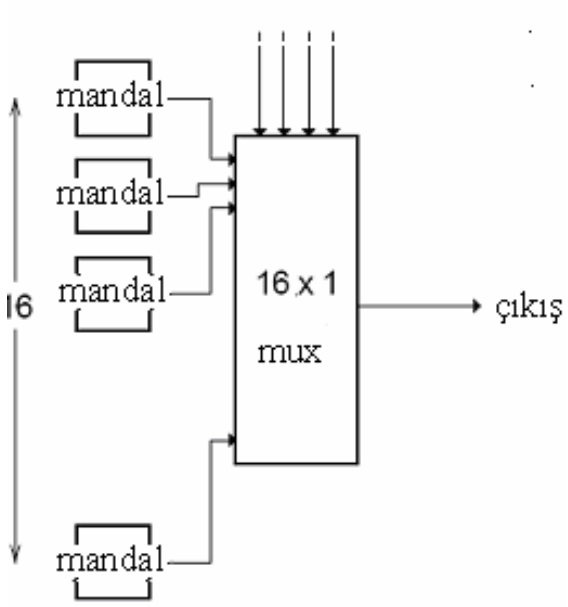


Şekil 3.3: FPGA iç blokları

Şimdi kısaca bu mimari hakkında genel bilgi verilecek.

Giriş Çıkış Blokları : Devrenin paket bağlantıları ile iç bağlantılar arasındaki ilişkinin oluşturulduğu kısımdır.

Düzenlenebilir Mantık Blokları: Mantık fonksiyonlarının gerçekleştirildiği alandır. Arama tabloları (look up table) ve flip floplar sayesinde çıkış işlemi gerçekleştirilir. Genel olarak FPGA içinde bulunan sistemler arama tabloları ile birbirinden ayrılır. 4 girişli arama tabloları tümleşik mantıksal işlemlerin yürütülmesini sağlar. Çıkış verisi de isteğe bağlı olarak yazmaca yazılır. Arama tabloları 2^nx1' lik bir hafıza işlemi görür. Gelen veriler 2^n tane hafıza birimlerinden birini seçer. Hafıza konumları kullanıcıdan gelen düzenleme bit dizisi ile dolar. MUX işlemi ile Düzenlenebilir Mantık Bloklarının giriş değerleri elde edilir. Bu sonuç olarak genel amaçlı mantıksal kapıya karşılık gelir.



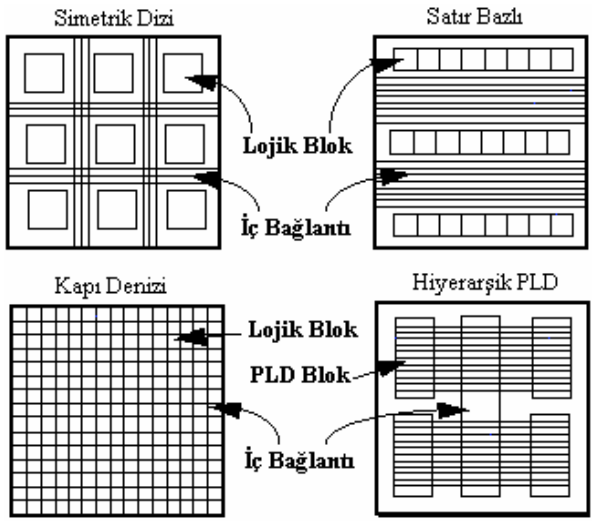
Şekil 3.4: Bit akış konfigürasyonun bir bölümü gibi programlanan mandallar

Boolen fonksiyonu kullanılarak mantık işlemleri gerçekleştirilir [24].

İç Bağlantı Blokları: Çoğu zaman Düzenlenebilir Mantık Bloklarının komşu bloklar ile birleşmesi gerekir. Her mantık bloğunun çıkışı ile bu blokların etrafındaki girişler arasında bağlantı oluşturulur. Bu bağlantı direk bağlantı olarak da adlandırılır. Bu komşu blokların yetersiz olduğu durumlarda ise genel bağlantılar (general

interconnect) kullanılabilir. Bu genel bağlantı hatları üzerinde anahtarlama devreleri (switched matrix, SM) olduğu için genel anlamdaki bağlantılar kolay bir şekilde gerçekleştirilebilir. FPGA'in uzak bölgelerinde ise uzun hatlar (long lines) kullanılabilir.

Şekil 3.5' de gösterildiği gibi FPGA' ler aynı zamanda bağlantı şekillerine göre simetrik dizi, sıra tabanlı dizi hiyerarşik dizi ve de kapı dizisi olmak üzere 4' e ayrılır.



Şekil 3.5. Bağlantı şekillerine göre FPGA'ler

3.6 Yüksek Hızlı Tümleşik Tanımlama Dili (Very High Speed Integrated Description Language , VHDL)

Yüksek Hızlı Tümleşik Tanımlama Dili (VHDL) 1987 yılında IEEE tarafından kabul edilen özellikle FPGA gibi mikroişlemcilerde kulanılabilen çok gelişmiş bir programlama dilidir. Tümleşik devrelerin kullanımının çok hızlı bir şekilde artması ve de özellikle programlanabilen devrelerde insanları ortak bir standart arayışına itmiştir [26].

VHDL dilinin diğer dillere göre birçok avantajı bulunur. Alt programlara ayrılarak daha iyi bir kullanım gerçekleştirilebilir. Ayrıca VHDL dili için Modelsim simülasyon programı kullanılarak devre tamamen tasarlanmadan sistem üzerinde denemeler yapılabilir.

Yapılan çalışmada üzerinde durulacak nokta öncelikle program yazıldıktan sonra devrenin gerçeklenip gerçeklenemediğinin Modelsim kullanılarak denenmesidir. Daha sonra eğer simülasyon yapılabiliyorsa, FPGA üzerine program yüklenerek FPGA sonuçları gözlenebilir. VHDL gelişmiş bir programlama dili olduğundan diğer dillerde yazılmış fonksiyonların da kullanımı mevcuttur. Ayrıca, tasarımın gerçeklenebilmesindeki kolaylık ve program üzerinde değişiklik yapabilme kabiliyeti de VHDL' in avantajları arasındadır [28].

Öncelikle VHDL ile ilgili olarak temel bilgiler verilecek. VHDL' i modüler olarak düşünmemiz bu programlama dilini daha iyi anlamamızı sağlayacaktır. Özellikle tezin algoritmaların gerçeklenmesi bölümünde modüler kısımların nasıl gerçekleştirildiği hakkında kapsamlı bir bilgi verilecektir [29].

Gerçeklenilen sistemde cdma2000 formatına uygun IEEE 754 kayan noktalı sayı formatı olduğundan dolayı çarpma ve toplama işlemleri dikkatli bir şekilde ele alınıp, ana program içerisinde gerekli zamanlar içinde çağırılmıştır [30]. Bu yüzden sistem iyi bir şekilde etüd edildikten sonra tasarlanarak giriş çıkış değerleri kontrol edilmelidir.

Şimdi VHDL yapısı hakkında bilgi verilecek. İlk olarak VHDL mimarisini oluşturan elemanlar hakkında bilgi verilecek. Varlık (entity), mimari (arhitecture), paket (package)(alt program) olarak bu yapılar sırasıyla incelenilecek.

Varlık (entity): Tasarımlar varlıklar ile ifade edilir. Varlıkların giriş çıkışlarına ise port adı verilir. Port tanımlamaları yapılan tasarıma göre farklı değerler alabilir. 'in' terimi yalnızca okunabilen değerleri ifade ederken, tasarımda giren eleman değerleri olarak kullanılır. Ve de çıkış işlemi olarak kullanılamaz. Genel olarak giriş değerleri çıkışta kullanımayacaksa tercih edilir. 'out' yalnızca çıkıştaki değerin portlara aktarılması için kullanılır. 'inout' bu değer ise döngülerde veya elde ettiğimiz çıkış değerlerini tekrar kullanmak amacıyla tercih edilir [31].

Bu yapıya yazılan program örneğini verebiliriz.

```
entity ccorelator_cikisi is
generic(e : integer :=5;--8
                                    f : integer :=10;--23
                                    s : integer :=1;--1
                                    bias : integer := 10;
                                    sifir:std_logic_vector(15 downto 0):="0000000000000000");
port(
                aci1_reel : out std_logic_vector(s+e+f-1 downto 0);
                aci1_im : out std_logic_vector(s+e+f-1 downto 0);
                clk : in std_logic
.....kod devam ediyor.....
 );
end ccorelator_cikisi;
```

Yukarıdaki örnekte yazılan SCC kodunun ana modülü üzerinde varlık kısmı gösterildi. Varlık kısmı içinde generic adı verilen kodda değişiklikler rahat bir şekilde yapılmasını sağlayan bölüm ve de port kısmı yer almıştır.

Mimari (Architecture) : Adından da anlaşıldığı gibi programın yapısının belirlendiği kısımdır. Tanımlanan programın temel olarak bu bölümde çalışması gerçeklenir.

```
architecture Behavioral of scarpma is
signal   RSy1RSy2:std_logic_vector(s+e+f-1 downto 0);
signal   RSy1ISy2:std_logic_vector(s+e+f-1 downto 0);
signal   ISy1RSy2:std_logic_vector(s+e+f-1 downto 0);
signal   ISy1ISy2:std_logic_vector(s+e+f-1 downto 0);
function carpma(sayi1,sayi2 : std_logic_vector (s+e+f-1 downto 0)) return std_logic_vector is
                variable s1,s2,ssonuc :std_logic;
                variable e1,e2,esonuc :std_logic_vector(e-1 downto 0);
                variable f1,f2,fsonuc :std_logic_vector(f-1 downto 0);
                variable v1,v2      :std_logic_vector(f downto 0);
                variable mc   :std_logic_vector(2*f+1 downto 0);
.....kod devam ediyor.....
end Behavioral;
```

Burada ise bir alt modül olan scarpma işleminin mimari (architecture) örnek olarak gösterilmiştir.

Paketler ve Altprogramlar: Programı yazarken kullanılacak alt programlar ve de veri tipleri bu yapıyla kullanılır. Özellikle daha önce de belirtildiği gibi IEEE 754 kayan noktalı sayı formatındaki toplama ve çarpma gibi işlemler alt program olarak belirlenmiştir. Alt programdan elde edilen çıkış değerleri yine ana programda değerlendirilir [31].

Bu bölümde paket olarak bir modül oluşturmaktan ziyade altprogramı çağıracak esas programda 'component ' olarak tanımlamalar yapıldı. Alt program olarak ise 16 veya 32 bitlik (generic) sanal çarpma ve toplama işlemini yapabilecek scarpma ve stoplama modülleri, sekizlik matris çarpma işlemi yapabilen america sekiz ve de 64'lük matris çarpma işlemi yapabilen ccorelator modülleri alt program olarak tanımlanmıştır. İlk olarak component yapısı hakkında bilgi verelim. Burada temel konu esas programdan alt modülün doğru bir şekilde çağrılabilmesidir.

```
component scarpma
port(   RSy1    : in std_logic_vector(s+e+f-1 downto 0);--:= "0011110000000000";-- 2.5
        ISy1    : in std_logic_vector(s+e+f-1 downto 0);--:= "0011110000000000";-- 3
        RSy2    : in std_logic_vector(s+e+f-1 downto 0);--:= "0011110000000000";-- 2.5
        ISy2    : in std_logic_vector(s+e+f-1 downto 0);--:= "0000000000000000";-- 3
        RCks    : out std_logic_vector(s+e+f-1 downto 0);
        ICks    : out std_logic_vector(s+e+f-1 downto 0);
        clk   : in  std_logic);
end component;

component stoplama
port(   RSy1    : in std_logic_vector(s+e+f-1 downto 0);    --:= "0100000100000000";-- 2.5
        ISy1    : in std_logic_vector(s+e+f-1 downto 0);    --:= "0100001000000000";-- 3
        RSy2    : in std_logic_vector(s+e+f-1 downto 0);    --:= "0100000100000000";-- 2.5
        ISy2    : in std_logic_vector(s+e+f-1 downto 0);    --:= "0100001000000000";-- 3
        RCks    : out std_logic_vector(s+e+f-1 downto 0);
        ICks    : out std_logic_vector(s+e+f-1 downto 0);
        clk     : in  std_logic);
end component;
```

Burada kullanılan alt programlardan scarpma ve stoplama' nın component kısmı örnek olarak gösterilmiştir. Böylece alt programın giriş ve çıkış değerleri esas programda belirlenmiştir.

```
M11M21carpma : scarpma
    portmap(matris1_reel1,matris1_im1,matris2_reel1,matris2_im1,carpim1_reel,carpim1_im,clk);
```

Yukarıdaki kodda ise alt programdan çagrılan bir sanal çarpma işlemi verilmiştir.

3.7 VHDL Nesne Türleri

VHDL nesnel özelliklere sahip olan bir dildir. Nesnelerin davranış ve işlevlerine göre fonksiyonellik kazanır. Nesneler modüler programlamaya uygun oldukları için yüksek performansa sahip yazılım dillerinde VHDL kullanılır. Şimdi nesne türlerinden sırasıyla bahsedilecek.

İşaret (signal): varlıkları (entity) bağlayan veri tipleridir. İşaretler (signal) varlık, mimari ve de paket içinde kullanılabilir. İşaretlerin paket içinde kullanımı çok önemlidir. Çünkü paket genel bir ifade olduğundan dolayı işaretler burada kullanıldığı zaman diğer varlık veya mimariler tarafından çağrılarak kullanılabilir. İşaretlere başlangıç değeri atanabilir. Program döndüğü zaman bu ifade yenilenebilir.

SIGNAL_signalin adı:signalin_tipi [:= başlangıç değeri]

```
signal lUT_reel11 : std_logic_vector(s+e+f-1 downto 0):=     "0011110000000000";
signal lUT_im11  : std_logic_vector(s+e+f-1 downto 0):=     "0011110000000000";
signal lUT_reel12 : std_logic_vector(s+e+f-1 downto 0):=     "0011110000000000";
signal lUT_im12  : std_logic_vector(s+e+f-1 downto 0):=     "0011110000000000";
signal lUT_reel13 : std_logic_vector(s+e+f-1 downto 0):=     "0011110000000000";
```

Örnek olarak yine yazdığımız programdaki arama tablosuna (look-up-table) ait herbir antene ait olan 90 açı değerinden 5 anten için 5x90 (450) açı değerinden ilk antene ait olan 5 açı değeri verilmiştir.

Değişkenler (variable): Genel olarak geçici değerleri kullanmak için tercih edilir. Böylece programın daha hızlı çalışması sağlanabilir. Fakat gerçek zamanlı işlemlerde özellikle tasarımın FPGA' de gerçeklenmesinde değişken kullanımının sakıncaları bulunmaktadır. Değişkenler tasarımda kullanıldıktan sonra kaldırıldığından FPGA üzerinde gerçek uygulamalarda kapı gecikmeleri yüzünden olumsuz durumlar oluşur. Bu sebeple yazılan kodda daha çok işaret (signal) kullanılmasına önem gösterildi. Bu gibi durumlar göz önüne alındığında process ve altprogram içinde geçici değerlerin tutulması faydalı olacaktır [32].

```
_değişkenin adı:değişkenin_tipi [:= başlangıç değeri]
        variable s1,s2,ssonuc :std_logic;
        variable e1,e2,esonuc :std_logic_vector(e-1 downto 0);
        variable f1,f2,fsonuc :std_logic_vector(f-1 downto 0);
        variable v1,v2        :std_logic_vector(f downto 0);
        variable mc           :std_logic_vector(2*f+1 downto 0);
        variable re           :std_logic_vector(e-1 downto 0);
        variable cikis        :std_logic_vector(s+e+f-1 downto 0);
```

Sayılan özelliklere rağmen iyi bir plan yapılarak değişkenlerin kullanımı hız açısından faydalı olabilir. Çünkü değişkenler işaretlere göre çok daha hızlıdır. Aynı zamanda işlemleri bittikten sonra ortadan kaybolur. Bir de işaretler daha fazla bilgi içerdikleri için daha fazla yer kaplayacakdır. Bu da genel olarak tasarımın çok fazla yer kaplaması anlamına gelecektir.

Sabitler (constant): Eğer tasarımda değişmez verilere ihtiyaç duyulursa sabitler kullanılır. Bu tasarımın daha iyi gözlemlenebilmesi ve de anlaşılabilmesi için önemli bir faktördür. Özellikle tasarım içinde sık sık kullanılan değeri aynı olan ifadeler için vazgeçilmez bir veri tipidir. Tabi sabitlerin de kullanımında işarette olduğu gibi beirli sınırlamalar olması gerekir. Aynı şekilde işaretlerde olduğu gibi paket kullanımı genel olarak düşünülmelidir. Eğer yalnızca bir process tarafından belirlenmişse yalnızca o process tarafından kullanılır [31].

VHDL çok geniş bir yapıya sahip olduğundan dolayı bir çok veri tipi kullanımı mevcuttur. Bu yüzden işaret ve sabit için söz konusu olan özellikler değişik veri tipleri içinde söz konusudur.

Burada örnek olarak tamsayılar kullanılabilir. Bu tamsayılar ile matematiksel işlemler çok rahat bir şekilde gerçekleştirilir.

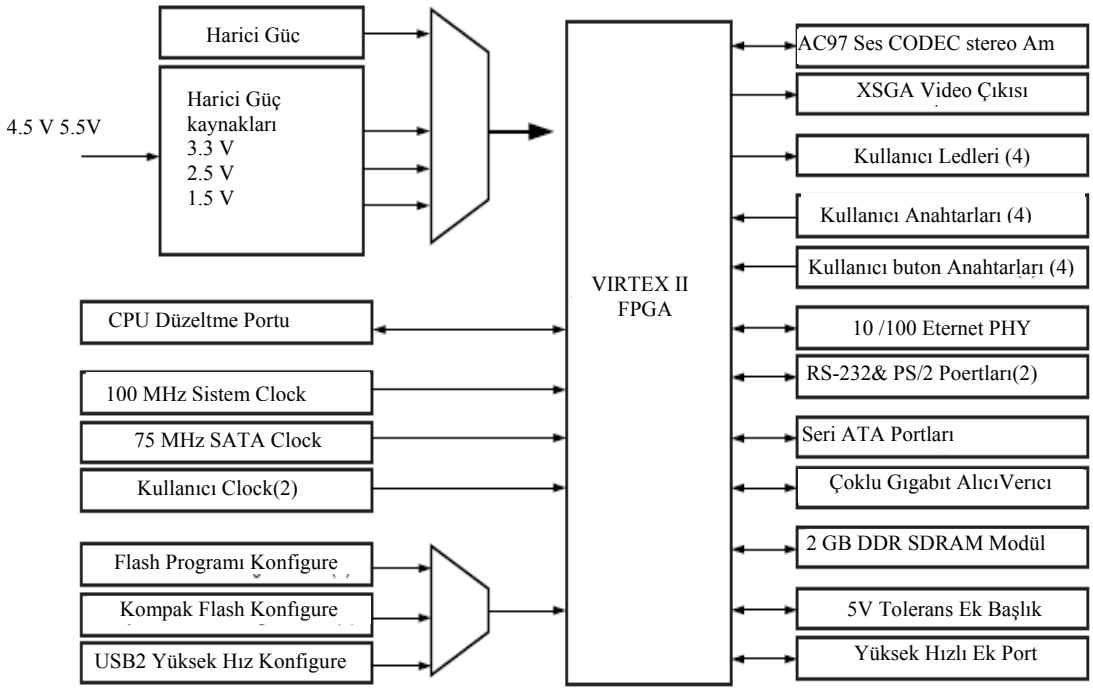
Diğer önemli bir tip ise gerçel sayılardır. Tam sayıda olduğu gibi belirli aralıklarda gerçel sayıların bir sınırlaması bulunur. Özellikle gerçek sayılar virgüllü ifadeler olduğu için virgülden sonra bir sayı belirtilmelidir. Sayı tamsayı ise virgülden sonra sıfır konulur.

Sıralı sayılar: veri tipleri tasarımcı tarafında oluşturulabilir. Bu yüzden tasarım sayı olarak değil de karakter olarak belirlenmelidir.

3.8 XUP Virtex II Geliştirme Sistemi

Bu bölümde kullanılan Xilinx ailesine ait XUP Virtex-II Geliştirme Sistemi üzerinde dururalacak. Öncelikle kodlanan algoritmaların simülasyonunu Modelsim simülasyon programı ile gerçekleştirilir. FPGA' e verinin yüklenmesi zaman alabileceğinden kod üzerinde değişiklikler yapabilmek için Modelsim esnek bir yapı sağlar. Bu bölümde veriyi aktardığımız XUP Virtex II Geliştirme Kartı özelliklerinden bahsedilecek [32]. Ayrıca algoritmalar Xilinx ailesine ait Virtex4 FPGA' i de kullanılarak gerçekleştirildi. Fakat katalog bilgisi olarak benzer özelliklere sahip yalnızca Virtex II verildi.

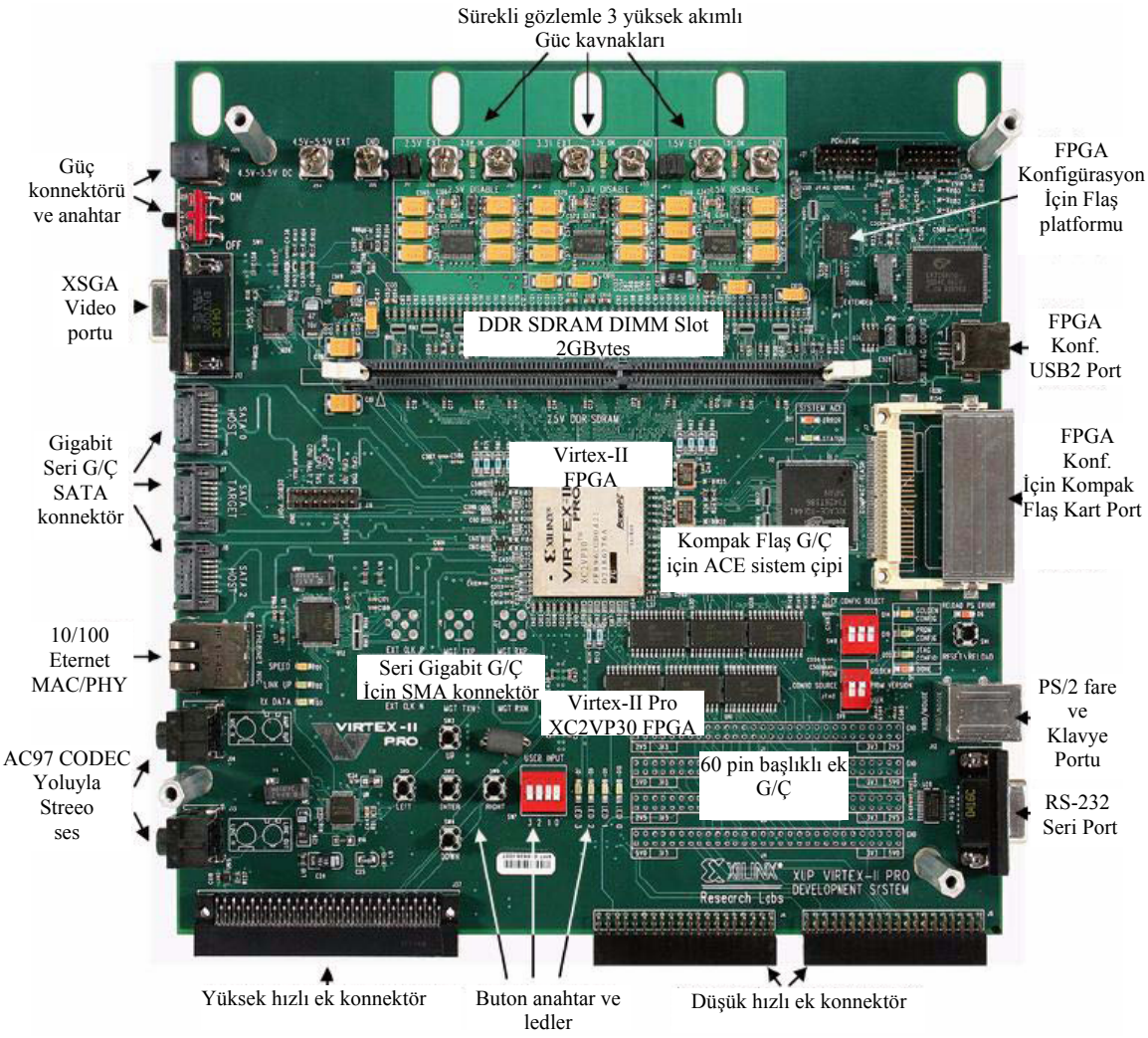
Öncelikle bu kartın fiziksel mimarisi hakkında bilgi verilecek. Şekil 3.6' da kullanılan Xilinx ailesine ait Virtex II FPGA' in fiziksel mimarisi gösterilmiştir.



Şekil 3.6. FPGA fiziksel mimarisi

3.8.1 Bord özellikleri

Bu bölümde XUP Virtex –II Pro Geliştirme Sistemi üzerinde birkaç önemli devre elemanı üzerinde durulacak. Şekil 3.7' de Virtex II FPGA bordu üzerinde yer alan kısımlar gösterilmiş olup, daha sonraki kısımlarda bu bölümler açıklanmıştır.



Şekil 3.7. FPGA bordu[32]

3.8.2. Güç kaynakları ve FPGA konfigürasyonu

XUP Virtex-II Geliştirme Sistemi 5V düzenli güç kaynağı ile beslenir. Bord üzerinde anahtarlanan güç kaynakları 3.3V, 2.5V ve 1.5V gerilim ile FPGA'i besler. Bu bord FPGA sayısal güç kaynaklarının tümü için mevcut ölçümleri sağlayabilir. Harici güç uygulamaları buna örnek olarak gösterilebilir. XUP Virtex-II Geliştirme Sistemi bord üzerinde konfigüre edilebilmek için birkaç metod içerir. Veri paralel port arayüzü veya gömülü platform kablosu olan USB' den iletilen harici konfigürasyon, depolama ortamı (8 potansiyel konfigürasyon) dahili kompak Flaş, dahili platform flaş PROM'dan üretilir [32].

3.8.3 Çoklu-gigabit alıcı verici

Sekiz Çoklu Gigabit alıcı verici (MGTs) XUP Virtex-II Geliştirme Sistemi içinde sunulmuştur. Bağlantı ve kullanıcılar tarafından yararlanılması için kullanılır. İki yönlü Çoklu Gigabit kanallarının üçü Seri Gelişmiş Ekleme Teknolojisi (Serial Advanced Technology Attachment, SATA) konnektörlere sonlandırılır. Dördüncü kanal kullanıcıdan kaynaklanan alt minyatür A (Sub Miniature A, SMA) birimine yönlendirilir [32].

3.8.4 RAM sistemi

XUP Virtex-II Pro Geliştirme Sistemi Çift Eş Zamanlı Dinamik Veri Oranı (Double Data Rate Synchronous Dynamic, DDRSD) RAM hafıza modülü içinde kullanıcı kaynaklı JEDEC standardı 184 çiftli pin yüklenmesi sağlanır. Bu bord 64 bit veya 72 bitlik veya daha az bit oranları ile 2Gb kapasiteli tamponlu veya tamponsuz hafıza modülünü destekler. Eğer hata tahmini veya doğrulaması gerekli ise 72 bit kullanımı uygundur.

3.8.5 Sistem geliştirme konfigürasyon ortamı denetleyicisi

Sistem Geliştirme Konfigürasyon Ortamı denetleyicisi FPGA konfigürasyon verisini yönetir. Bu denetleyici bir FPGA hedef zinciri ve çeşitli desteklenen konfigürasyon kaynakları arasında akıllı bir arayüz sağlar. XUP Virtex-II Geliştirme Sistemi tekli sistem denetleyicisi sağlar [32].

3.8.6 Hızlı ethernet ara yüzü (Fast ethernet interface)

XUP Virtex-II Geliştirme Sistemi IEEE Ethernet alıcı-vericisi sağlar. Bu 100 BASE-TX ve 10BASE-T uygulamalarını destekler. Bu otomatik devir ve paralel tahmin ile birlikte 10Mbs ve 100Mbs çift tam işlemi destekler [32].

3.8.7 Seri port

XUP VirtexII geliştirme sistemi tekli RS-232 portu ve çift PS2 portları olarak üç seri porta bağlanır. RS-232 portu bir DB-9 seri konnektör kullanarak el sıkıştırma donanımıyla bağlanır [32].

3.8.8 Kullanıcı ledleri ve anahtarlar

LED'lerin tümü kullanıcı tanımlı amaçlı olarak kullanılır. FPGA lojik '0'olarak sürüldüğü zaman, buna karşılık LED' ler yanar. Ayrıca 4 pozisyonlu anahtarların kullanımı mevcuttur [32].

3.8.9 Yayılım konnektörleri (Expansion connectors)

XUP Virtex-II Geliştirme Sistemi giriş çıkış pinlerinin tümü kullanıcı belirlemek için 40 pin sağ açı konnektörü ve 4 kullanıcı kaynaklı 60 pin başlıktan oluşur [32].

3.8.10 XSGA çıkışı

XUP Virtex-II Geliştirme Sistemi bir video sayısal analog dönüştürücü (Digital Analog Converter, DAC) ve XSGA çıkışı sağlamak için 15 pin yüksek yoğunluklu D alt konnektör içerir [32].

3.8.11 USB2 programlama arayüzü

XUP Virtex-II Geliştirme Sistemi 480Mbs yüksek hıza sahip veya tam hızlı 12 Mbs haberleşme yeteneğine sahip olan bir gömülü USB 2.0 mikrodenetleyici içerir. Bu arayüz programlama ve konfigürasyon işlemleri için önemli bir arayüz sağlar [32].

Bu bölümde, genel olarak FPGA'ler, FPGA'leri proglamlamak için kullanılan VHDL dili bir de algoritmaların gerçeklenmesinde kullandığımız Xilinx ailesine ait Virtex II FPGA hakkında bilgi verildi. Aynı zamanda, yapılan çalışmada Virtex4 FPGA kullanıldı. Fakat benzer özelliklere sahip olması sebebiyle Virtex2 FP GA

anlatıldı. Genel olarak virtex4 daha iyi kapasite özelliklerinin yanında içerisinde bulunan DSP ile karmaşık algoritmaların gerçeklenmesi açısından daha iyi sonuçlar ortaya koyar.

BÖLÜM 4. İŞARETİN GENEL KARAKTERİSTİĞİ VE KULLANILAN AKILLI ANTEN ALGORİTMALARI

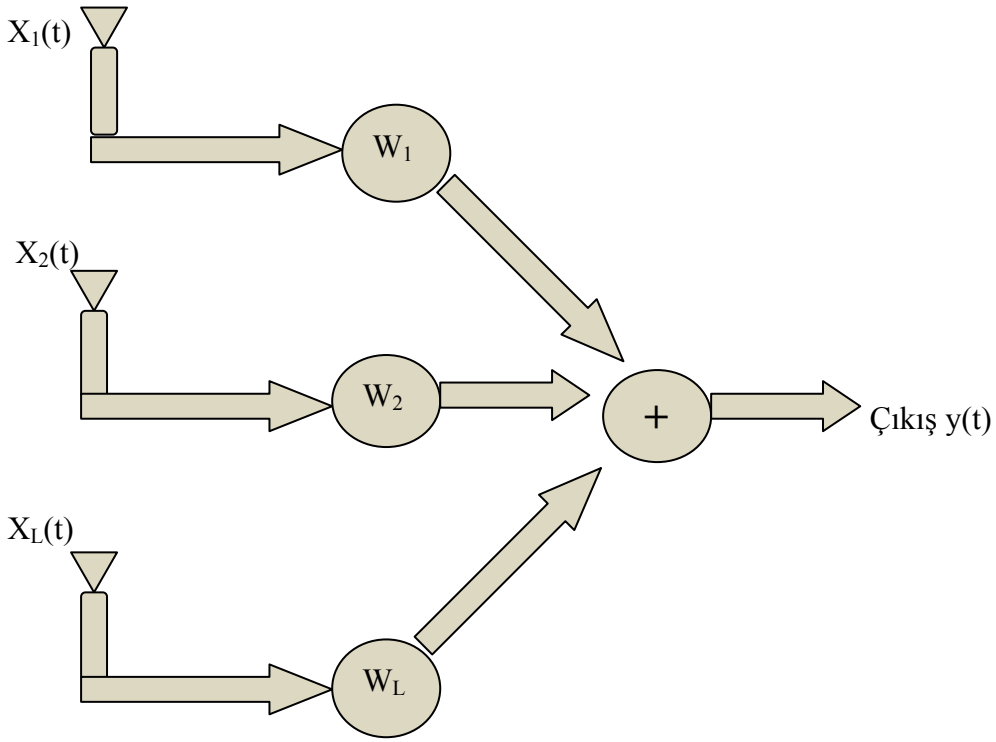
4.1 Giriş

Bu bölümde, işaretin genel olarak karekteristiği ve akıllı anten algoritmaları hakkında bilgi verilecek. Akıllı antenler, gelişmiş işaret işleme mekanizmalarına sahip oldukları için matematiksel olarak işaret modeli de önem kazanmaktadır. Bu sebeple, basit bir işaret modeli ile başlanarak işaretin genel modeli verildikten sonra, gürültü girişim ve çokluyol etkileri göz önüne alınarak işlem genişletilecek ve de ışın şekillendirme, işaret-gürültü oranı, işaret-girişim-gürültü oranı gibi kavramlar matematiksel olarak ifade edilecektir.

Bir sonraki kısımda ışın şekillendirme için kullanılan akıllı anten algoritmaları üzerinde durularak, yapılan çalışmada kullanılan En Küçük Karesel Ortalama (Least Mean Square, LMS), Sabit Modül (Constant Modülüs, CM) ve Uzaysal Kod Korelatör (Space Code Correlator, SCC) Algoritmaları geniş bir şekilde anlatılacaktır.

4.2 Genel İşaret Modeli

L elemanlı bir anten dizisine sahip olunduğunu düşünelim. Yapılan çalışmada L elemanı sayısı 5’ e karşılık gelir. Anten dizilerine gelen işaretlerin herbiri uygun ağırlık vektörleri ile çarpılarak toplamında tek bir işaret elde edilir. Bu elde edilecek işaret çeşitli kriter değerler ile karşılaştırma işlemi yapılarak, kullanılacak akıllı anten algoritmaları ile iterasyon işlemi yapılır. Sonunda istenen işaret minimum hata ile elde edilmeye çalışılır. Başlangıçta genel olarak çıkış işaretinin elde edilmesini incelenilecek. Daha sonra ise yapılan ışın şekillendirme ve işaretin diğer özelliklerinin kapsamlı bir şekilde ele alınacak.



Şekil 4.1 Anten dizisi

$$y(t) = \sum_{i=1}^{L} \mathbf{w}_1^* \mathbf{x_1}(t) \tag{4.1}$$

Burada * işareti, karmaşık eşlenik olduğunu göstermektedir. Karmaşık değerlerin eşleniği, matematiksel gösterim kolaylığı için kullanılmaktadır.

Dizi sisteminde ağırlıklar vektör formunda aşağıdaki gibi gösterilmektedir.

$$\mathbf{w} = [w_{1,} w_2 .....w_L]^T \tag{4.2}$$

Her bir anten elemanından elde edilen işaretlerde

$$\mathbf{x} = [x_1(t) x_2(t) .....x_L(t)]^T \tag{4.3}$$

şeklide gösterilirse, dizi çıkışındaki toplam işaret,

$$y(t) = \mathbf{w}^H \mathbf{x}(t) \tag{4.4}$$

olarak elde edilir. Denklemlerde kullanılan T ifadesi, matrisin yada vektörün devriğini (transpozesini) ve H ise karmaşık eşlenik devriğini ifade etmektedirler. **w** ağırlık vektörünü, **x**(t) ise işaret vektörünü göstermektedir.

Dizi çıkışında her bir anten elemanından alınan işaretler ilgili ağırlık ile çarpıldıktan sonra toplanmaktadır. 4.4 denkleminde **w** ağırlık vektörünü, **x**(t) ise işaret vektörünü gösterdiği için, bu durum iç çarpım ile sağlanmaktadır.

Dizi çıkışının herhangi bir t anında gücü, çıkış işaretinin karesinin alınmasıyla elde edilebilir.

$$P(t) = \left| y(t) \right|^2 \tag{4.5}$$

$$P(t) = y(t) y(t)^*$$

(4.4) denkleminden y(t) yerine yazılırsa,

$$P(t) = \mathbf{w}^H \mathbf{x}(t) \mathbf{x}^H(t) \mathbf{w} \tag{4.6}$$

ifadesi elde edilir.

Eğer x(t) bileşenleri, ortalaması sıfır olan durağan bir işlem olarak modellenebilirse, verilen bir **w** ağırlık vektörü için dizi sisteminin ortalama çıkış gücü, **x**(t) üzerinden koşullu beklendik değerinin alınmasıyla elde edilebilir [33]. Bu durumda güç

$$P(t) = E[\mathbf{w}^H \mathbf{x}(t) \mathbf{x}^H(t) \mathbf{w}] \tag{4.7}$$

$$P(t) = \mathbf{w}^H E[\mathbf{x}(t) \mathbf{x}^H(t)] \mathbf{w} \tag{4.8}$$

4.3 Adaptif Dizi İşlemleri

İşaret alıcı verici arasında gidip gelirken, çeşitli çokluyol etkilerinden dolayı saçılır, yayılır ve yansır. Buna ek olarak kullanıcının hareketi taşıyıcı frekansı içinde doppler kaymasına sebep olur. Sonuç olarak bu işaretler alıcıda birleştiğinde zayıflama ve faz kaymasına sebep olur. Gezgin haberleşme performansını azaltan diğer önemli bir faktör ise girişimdir. Bu bölümde sistemin performansını iyileştiren ve geliştiren yöntemler üzerinde durulacaktır. Uzaysal teknikler olup, çeşitlilik ve adaptif ışın şekillendirici olarak isimlendirilir. Çeşitlilik teknikleri sönümleme gibi etkileri ortadan kaldırmak için kenar boşluğu içinde kazanç sağlar.

Sayısal haberleşme sistemleri içinde gerekli SNR veya E_b/N_0 içinde geliştirilen bu sonuçlar Bit Hata Oranı (BER) terimleri içinde verilen servis kalitesini başarmak için gereklidir [34].

Benzer olarak ışın şekillendirme dizisinin kazanç terimleri, girişim azaltımı ve uzaysal filtreleme içinde gerekli birkaç çeşitlilik sağlar. Yapılan çalışmada yalnızca yukarı bağlantı ile ilgilenildiğinden bu kısımda yalnızca yukarı bağlantı incelenecektir.

4.3.1 Çeşitlilik teknikleri

Gezgin kullanıcı yavaş ve hızlı sönümleme etkilerine sahip olabilir. Yavaş sönümlemenin etkileri bütün cdma sitemleri durumunda kapalı döngü güç kontrolü kullanılarak kolaylıkla üstesinden gelinebilir. Güç kontrolü hızı cdma2000 sistemlerde 800 Hz ve WCDMA sistemlerde 1500 Hz olarak aşağı bağlantı ve yukarı bağlantı üzerinde kullanılır. Hatta bu hızlarda bile güç kontrolü hızlı sönümlemeye sahip gezgin istasyonda yeterli bir hız olmayabilir. Alış çeşitlemesi özellikle Rayleigh sönümleme içinde hızlı sönümlemeye karşın alıcının bağışıklığını arttırmak için daha etkili bir çözümdür.

Eğer farklı çokluyollardan gelen işaret iyi bir şekilde ilişkilendirilmişse, işaretle birlikte sönümlenebilirler ve de hiçbir şekilde çeşitlik sağlanamamış olunur. Çoklu

yollar arasındaki çapraz korelasyon katsayısı 0.7 veya daha az olursa, iyileştirme kabul edilebilir [22]. Eğer çokluyolların ortalama işaret gücü büyük ölçüde eşit değil ise veya dengelenmemiş ise bu çeşitlilik birleştirme tekniklerinin performansını etkileyebilir.

Çeşitlilik kazancı özel bir olasılık fonksiyonu olarak alıcı çıkışında sönümlenen işaretin Birikimli Dağılımı (Cumulative Distribution) fonksiyonu ile karşılaştırılarak genel olarak hesaplanır. Deneysel olarak elde edilen bu çeşitlilik kazancı şu şekilde ifade edilir. Maksimum oranlı birleştirme için kısaca % 90 işaret güvenirliği 2 dilimli çeşitlilik kazancı

$$G_{div}=7.14e^{(-0.59\rho-0.11\Delta)} \tag{4.9}$$

Burada ρ işaretler arasındaki çapraz korelasyonu ifade ederken, Δ ortalama işaret seviye farkını gösterir [35].

4.3.2 Açı çeşitliliği

Birçok anten şeması, uzaysal ve kutupsal çeşitlilik içeren, yüksek kazanç başarmak için gerekli kanal çeşitliliğini oluşturur. Uzaysal çeşitlilik içinde, kablosuz haberleşme sistemleri içinde popüler bir teknik olarak kullanılır. Uzay çeşitli sistemler, ilişkilendirilmemiş işaretler için başarılan maksimum kazanç ile düşük çapraz korelasyona sahip alıcının farklı antenlerinde tasarlanır.

Açı çeşitliliği şehir ve şehir gibi yoğunluğun fazla olduğu yerlerde önemli kazanç sağlar. Bunlar zengin çokluyolları ortaya çıkarır. Şehir ve şehir gibi alanlarda çok fazla miktarda yayılma meydana gelir. Gezgin istasyon işareti birçok DOA ile temel istasyonda alınır. Böylece, açı çeşitliliği ile kısmen çokluyolların üstesinden gelinir.

Diğer yandan temel istasyonda alınan işaret kırsal ortam içinde LOS parçaları ve çok fazla çokluyol elemanı ile bozulabilir. Bu sebeple, temel istasyon içinde kullanılan darband ışınlar çoklu DOA' lardan gelen işaretin farklı kopyalarını yakalamak için alıcıya izin verir. Daha dar bir ışın zengin çokluyol ortamı içinde daha iyi bir çözüm

sunar. Açı çeşitliliği uzay çeşitliliği ile birlikte karşılaştırılabilir çeşitlilik kazancı sağlar. Bu zengin yayılma kanalından bütün ışınlar karşılaştırılabilir ortalama güç ve düşük çapraz korelasyon matrisiyle alınan işareti belirler [35].

4.3.3 Maksimum oran birleşimi

Çoklu anahtarlamalı ışın sistemi, bütün portlar içinde işaretlerin bir sayısını üretir. Işın içinde elde edilebilir bütün yollardan işaretlerin kullanımı gerçekleştirilir. Çeşitlilik işlemi, bütün bu işaretleri birleştirmek için kullanılabilir.

Klasik anahtar ve seçici çeşitlilik, bütün mümkün çeşitli yolların dışında bir işaret seçer. Bunlar elimizde bulunan uygulamalar için uygun değildir. Diğer yandan eşit kazanç birleştirme teknikleri bütün işaretlere farklı faz kaymaları uygular ve onları eşit genliklerde birleştirir. Bu tekniğin bir dezavantajı dilimlerden biri, diğer dilimlere göre oldukça düşük bir SNR değerine sahipse, bütün sistemin SNR değeri azalır. En uygun çeşitlilik şeması MRC' dir. MRC içinde alınan işaretler ilk komşu fazlardır, anlık SNR'lar ile ağırlıklandırılır ve sonra birleştirilir [33].

Farzedelim ki L çeşitlilik dilimine sahibiz, i. dilim tarafından alınan işaret

$$x_i = sh_i + n_i \tag{4.10}$$

ile gösterilir. s iletilen işareti ve n_i eklenmiş gürültüyü gösterir.
Her dilim için optimum ağırlık vektörü,

$$w_i = \frac{h_i}{\sigma_i^2} \tag{4.11}$$

İle ifade edilir. σ_i^2 dilim için gürültü gücünü ifade eder.

Birleştirilen çıkış için SNR,

$$SNR_{MRC} = \frac{\left|\sum_{i=1}^{L} w_i h_i\right|^2}{\sum_{i=1}^{L} w_i^2 h_i^2} \tag{4.12}$$

ile gösterilir. Tek dilimdeki gürültü gücü σ_i^2'ye eşittir.

$$SNR_{MRC} = \frac{\left|\sum_{i=1}^{L} \frac{h_i^2}{\sigma_n^2} h_i\right|^2}{\sum_{i=1}^{L}\left(\frac{h_i^2}{\sigma_n^2}\right)^2 \sigma_n^2} = \frac{1}{2}\sum_{i=1}^{L} \frac{|h_i|^2}{\sigma_n^2} = \sum_{i=1}^{L} SNR_i \tag{4.13}$$

ile ifadelenir [34].

4.3.4 Adaptif ışın şekillendirme

M elemandan oluşan doğrusal bir dizi düşünülürse, bu dizi $\{\theta_1, \theta_2 \ldots\ldots\ldots \theta_L\}$ doğrultularında dizi üzerinde ω_0 frekansında merkezlenen dalgaların sayısı L olsun. Karmaşık işaret kullanılarak m. eleman için alınan işaret,

$$x_m = \sum_{i=1}^{L} s_i(t) e^{-j(i-1)k_i} + n_m(t) \tag{4.14}$$

ile gösterilir. Burada $s_i(t)$ m. elemanda i. kaynağın işaretine karşılık gelirken, $n_m(t)$ ise m. elamanda alınan gürültü işaretini temsil eder.

$$k_i = \frac{2\pi d}{\lambda}\sin(\vartheta_i) \tag{4.15}$$

Vektör notasyonu kullanılarak matris formu içinde çıkış dizisi yazabiliriz.

X(t) =AS(t)N(t) (4.16)

Burada,

$$\mathbf{X}(t) = [x_1(t) \quad x_2(t) \quad .....x_M(t)]^T \tag{4.17}$$

$$\mathbf{S}(t) = [s_1(t) \quad s_2(t) \quad .....s_M(t)]^T \tag{4.18}$$

$$\mathbf{N}(t) = [n_1(t) \quad n_2(t) \quad .....n_M(t)]^T \tag{4.19}$$

Burada, T matrisin transpozesini gösterir. Yönlendirme matrisinin sütunları θ_i açsında işarete karşılık gelen yönlendirme vektörleridir.

$$\mathbf{A = [\ a(\theta_1) \quad a(\theta_2) \quad \ldots\ldots\ldots\ldots\ a(\theta_L)\ ]} \tag{4.20}$$

Doğrusal dizinin yönlendirme vektörü,

$$\mathbf{a}\ (\ \theta_i\ ) = [1 \quad e^{-jki} \quad e^{-j2ki} \ldots\ldots\ldots\ldots\ e^{-j(M-1)ki}\ ]^T \tag{4.21}$$

olarak belirtilir. Kablosuz haberleşme sistemlerinde bu dizi direk doğrultu boyunca alınan işarete ek olarak farklı DOA (Direction of Arrival)' lar ile birlikte iletilen işaretin çokluyol elemanlarını alır. Böylece toplam işaret vektörünü gürültüsüz ve girişimsiz olarak yazabiliriz.

$$\mathbf{a}(\boldsymbol{\theta}_1)s_1(t) + \sum_{l=2}^{L} \alpha_l \mathbf{a}(\boldsymbol{\theta}_1)s_1(t) = \mathbf{a}_1(\boldsymbol{\theta}_1)s_1(t) \tag{4.22}$$

Burada a_1 uzaysal imza olarak adlandırılırken, α_l direk yol ile l. eleman arasındaki faz ve genlik farkı olarak karşımıza çıkar. Adaptif diziler içinde karmaşık ağırlıklar M boyutlu çıkış elemanlarına uygulanır.

$$\mathbf{W=[w_1 \quad w_2 \quad .....w_M]}^T \tag{4.23}$$

Daha sonra çıkış dizisi,

$$\mathbf{y}(t)= \sum_{i=1}^{M} w_i^{*} x_i(t) = \mathbf{W}^{H}\mathbf{X}(t) \tag{4.24}$$

elde edilebilir. Ortalama çıkış gücü,

$$P(w)=E \mid y(t)\, y^{*}(t) \mid = \mathbf{W}^{H}\mathbf{R_s}\mathbf{W} + \mathbf{W}^{\mathbf{H}}\mathbf{R_N}\mathbf{W} \tag{4.25}$$

ile bulunur.

* e eşlenik anlamına gelirken, E [.] beklenen işareti ifade eder. R dizi korelasyonu

$$\mathbf{R}= E\,[\,\mathbf{X(t)}\,\mathbf{X^{H}(t)}] = \mathbf{R}_s + \mathbf{R}_N \tag{4.26}$$

ile ifade edilir.

Burada H bir matrisin veya vektörün eşlenik transpozesi anlamına, R_s işaretin korelasyon matrisi ve R_N gürültü ve girişimlerden oluşan korelasyon matrisini verir. Ağırlık vektörlerini en uygun hale dönüştürmek için birçok teknik ortaya atılmıştır. Daha sonraki bölümlerde bunlara değinilecektir [33].

4.3.5 Işın Yönlendirme

Işın yönlendirme bir toplama veya gecikme işlemi yapılarak ışın şekillendirici tarafından gerçekleştirilebilir. Işın şekillendiricide ağırlıklar genliğe bağlı olarak yayılımı gerçekleştirir. Öyle ki faz θ_0 kısmi doğrultusunda diziyi asıl işarete yönlendirmek için seçilir.

Bu dizi ağırlıkları,

$$\mathbf{W}_{\mathrm{BS}} = \frac{1}{\mathrm{M}}\mathbf{a}(\theta_0) \tag{4.27}$$

ile ifade edilir. Burada M eleman sayısını ifade ederken, $\mathbf{a}(\theta_0)$ yönlendirme vektörünü gösterir. Diğer yandan dizi asıl ışını istenen kaynağın DOA'sı boyunca

yönlendirir. Bu sebeple yönlendirme yapabilmek için istenen işaretin yön bilgisi gereklidir. Işın şekillendirici çıkışındaki SNR

$$\mathrm{SNR}_{\mathrm{BS}} = \frac{\mathbf{W}^{\mathrm{H}}\mathbf{R}_s W}{\mathbf{W}^{H}\mathbf{R}_N W} \tag{4.28}$$

ile ifade edilir.

Özel durumda bu sistem korelasyon yapılmadan gürültü baskın hale gelebilir. Bu durumda girişim ortaya çıkabilir.

$$\mathrm{SNR}_{\mathrm{BS}} = \frac{\frac{1}{M^2}\mathbf{a}^{H}(\theta_0)\mathbf{a}(\theta_0)\mathbf{A}P\mathbf{A}^{H}\mathbf{a}(\theta_0)\mathbf{a}^{H}(\theta_0)}{\frac{\sigma_{\mathrm{n}}^2}{\mathrm{M}^2}\mathbf{a}^{H}(\theta_0)\mathbf{a}(\theta_0)} = \frac{P}{\frac{\sigma_n^2}{M}} = \frac{MP}{\sigma_n^2} \tag{4.29}$$

4.3.6 Boşa yönlendirmeli ışın şekillendirici

Boşa yönlendirmeli ışın şekillendirici, bilinen bir doğrultudan alınan işaretin DOA'sına göre dizi cevabı içinde boşluk oluşturur. Böylece, girişimler önlenebilir. Bu ışın şekillendirme işlemi ağırlıkların seçimi yapılarak güncellenebilir. Bu yüzden girişim doğrultusunda boşluk oluşturularak birim kazanç ile ışın istenen işaretin DOA'sı boyunca üretilir.

$$\mathbf{W}^{\mathrm{H}}\ \mathbf{A} = \mathbf{W}^{\mathrm{H}}\ [\mathbf{a}(\theta_0)\,\mathbf{a}(\theta_1)\ldots\ldots\ldots\ldots\mathbf{a}(\theta_k)] = \begin{bmatrix}\mathbf{1}\\ \mathbf{0}\\ \mathbf{.}\\ \mathbf{0}\end{bmatrix} \tag{4.30}$$

Burada $\mathbf{a}(\theta_0)$ istenen işaret için yönlendirme vektörünü gösterirken, $\mathbf{a}(\theta_1)\ldots\ldots\ldots\ldots\mathbf{a}(\theta_k)$ girişim ile ilgili olan işaretleri gösterir.

Boş yönlendirme tekniklerinin bir dezavantajı bütün girişim kaynaklarının DOA bilgisini bilmesini gerektirmesidir. Ek olarak, özel durumlar içinde SNR maksimum yapan değerler için ışını boşluk doğrultusunda yönlendirme doğru sonuçlar vermeyebilir [6].

4.3.7 Maksimum işaret girişim ve gürültü oranı

İşaret gürültü oranını (SINR) aşağıdaki gibi belirtebiliriz.

$$\mathrm{SINR} = \frac{\mathbf{W}^{\mathrm{H}}\mathbf{R}_s\mathbf{W}}{\mathbf{W}^{H}\mathbf{R}_N\mathbf{W}} \tag{4.31}$$

R_S=E [S S^H] istenen işaretin korelasyon matrisin, R_N=E [N N^H] gürültü ve girişim işaretinin korelasyon matrisini ifade eder. Eğer $\mathbf{W}^{\mathrm{H}}\mathbf{R}_s\mathbf{W}$ değerini $\mathbf{W}^{H}\mathbf{R}_N\mathbf{W}$ =1 değeri ile maksimum yaparsak maksimum SINR değerini elde ermiş oluruz [35]. Lafrange çarpımı ve Gradient kümeleri ile aşağıdaki şekildeki gibi ifade edilebilir.

$$\nabla[\mathbf{W}^{\mathrm{H}}\mathbf{R}_s\mathbf{W} + \lambda(1 - \mathbf{W}^{H}\mathbf{R}_N\mathbf{W})] = \mathbf{R}_s\mathbf{W} - \lambda\mathbf{R}_N\mathbf{W} = 0 \tag{4.32}$$

Öz değer problemi içinde bu sonuçlar

$$\mathbf{R}_s\mathbf{W} = \lambda\mathbf{R}_N\mathbf{W} \tag{4.33}$$

elde edilir. Bir sonraki adımda ise,

$$\mathbf{R}_N{}^{-1}\mathbf{R}_s\mathbf{W} = \lambda\mathbf{W} \tag{4.34}$$

bulunur.

4.34' deki ifadeninin çözümü için $\mathbf{W}\ \mathbf{R}_N^{-1}\mathbf{R}_s$ 'nin bir öz vektörü olur. Böylece maksimum özdeğere karşılık gelen öz vektör SINR' ı maksimım yapabilir ve $SINR_{max}=\lambda_{max}$ olur. Böylece,

$$\mathbf{R}_s\mathbf{W}=\lambda_{\max}\mathbf{R}_N\mathbf{W}=SINR.\mathbf{R}_N\mathbf{W} \tag{4.35}$$

Optimum ağırlıklar ve işaretin korelasyon matrisinin tanımı kullanılarak,

$$\mathbf{W}_{opt}=K_{SINR}\mathbf{R}_N^{-1}\mathbf{A} \tag{4.36}$$

bulunur.

4.3.8 Optimum işaret girişim gürültü oranı

Daha önce $\mathbf{W}_{opt}=K.\mathbf{R}_N^{-1}\mathbf{A}$ formülünü elde etmiştik. A bir yönlendirme vektörü veya matrisini tercih edebilir veya uzaysal imza yerine yerleştirilebilir. Böylece optimum SINR optimum ağırlıklar yerine konularak şu şekilde elde edilebilir.

$$\text{SINR}=\frac{(K.\mathbf{R}_N^{-1}\mathbf{A})^H\mathbf{R}_s^{-1}(K.\mathbf{R}_N^{-1}\mathbf{A})}{(K.\mathbf{R}_N^{-1}\mathbf{A})^H\mathbf{R}_N^{-1}(K.\mathbf{R}_N^{-1}\mathbf{A})}=\sigma_s^2\mathbf{A}^H\mathbf{R}_N^{-1}\mathbf{A} \tag{4.37}$$

Örnek olarak, CDMA yukarı bağlantı kanalı için örneklenmiş çoklu oran sunulabilir. Burada düşük veri oranlı olarak ses kullanıcısı, yüksek veri oranlı kullanıcı ve bir beyaz gürültü terimi içeriyor. Diğer taraftan bir de yeterli düşük veriye sahip CDMA kullanıcıları sunulduğunu farz edelim. Bağımsız Rayleigh sönümlemesi beyaz uzaysal gürültü gibi modellenir. Her anten elemanı üzerinde beyaz gürültü eşitliği karmaşık gaussian ve sıfır ortalamalı varyans içerir.

$$\mathbf{X}(t)=\mathbf{A}_d S_d(t)+\mathbf{A_I} S_I(t)+\mathbf{N}(t) \tag{4.38}$$

Burada S_d istenen işarete ve S_I girişim işaretine karşılık gelir.

Farz edelim ki, beyaz gürültü terimi eşitliği S_d, S_l, A_d ve A_I ile ilişkilendirilemez. S_d işareti içindeki güç $E[S_dS_d^H]= \sigma_s^2$ ve S_l işareti içinde güç $E[S_IS_I^H]= \sigma_I^2$. Gürültü ve girişim korelasyon matrisi

$$\mathbf{R}_{NI} = \sigma_I^2 \mathbf{A}_I \mathbf{A}_I^H + \mathbf{R}_N \tag{4.39}$$

bulunur. Burada $\mathbf{R}_N = \sigma^2 \mathbf{I}$. Matris tersi kullanılarak aşağıdaki formül elde edilebilir. [35].

$$\mathbf{R}_{NI}^{-1} = \mathbf{R}_N^{-1} - \mathbf{R}_N^{-1}\sigma_I \mathbf{A}_I [1 + \sigma_I \mathbf{A}_I^H \mathbf{R}_N^{-1} \sigma_I \mathbf{A}_I]^{-1} \sigma_I \mathbf{A}_I^H \mathbf{R}_N^{-1} \tag{4.40}$$

$$\mathbf{R}_{NI}^{-1} = \frac{1}{\sigma^2}\left[I - \frac{\sigma_I^2}{\sigma^2}\mathbf{A}_I \mathbf{A}_I^H \frac{1}{\left(1 + \frac{\sigma_I^2}{\sigma^2}\mathbf{A}_I^H \mathbf{A}_I\right)}\right] \tag{4.41}$$

$$\mathbf{R}_{NI}^{-1} = \frac{1}{\sigma^2}\left[I - \frac{\mathbf{A}_I \mathbf{A}_I^H}{\left(\frac{\sigma^2}{\sigma_I^2}\mathbf{A}_I^H \mathbf{A}_I\right)}\right] \tag{4.42}$$

$$SINR_{opt} = \sigma_s^2 \frac{\mathbf{A}_d^H}{\sigma^2}\left[I - \frac{\mathbf{A}_I \mathbf{A}_I^H}{\left(\frac{\sigma^2}{\sigma_I^2}\mathbf{A}_I^H \mathbf{A}_I\right)}\right] \mathrm{A_d} \tag{4.43}$$

$$SINR_{opt} = \frac{\sigma_s^2}{\sigma^2}\left[A^H \mathbf{A}_d - \frac{\mathbf{A}_d^H \mathbf{A}_I \mathbf{A}_I^H \mathbf{A}_d}{\left(\frac{\sigma^2}{\sigma_I^2}\mathbf{A}_I^H \mathbf{A}_I\right)}\right] \tag{4.44}$$

$$A_I^H A_d = \| A_I^H \| \| A_d \| \cos(\theta_s) \tag{4.45}$$

Burada θ_s girişim ve işaret yönlendirme vektörleri arasındaki açıyı gösteriyor.

$$SINR_{opt} = \frac{\sigma_s^2 \left\| \mathbf{A}_d \right\|^2}{\sigma^2} \left[1 - \frac{\left\| \mathbf{A}_I \right\|^2 \cos^2(\theta_s)}{\left(\frac{\sigma^2}{\sigma_I^2} + \left\| \mathbf{A}_I \right\|^2 \right)} \right] \tag{4.46}$$

$$SINR_{opt} = \frac{\sigma_s^2 \left\| \mathbf{A}_d \right\|^2}{\sigma^2} \left[\frac{\left\| \mathbf{A}_I \right\|^2 \sin^2(\theta_s) + \frac{\sigma^2}{\sigma_I^2}}{\left(\frac{\sigma^2}{\sigma_I^2} + \left\| \mathbf{A}_I \right\|^2 \right)} \right] \tag{4.47}$$

4.3.9 Yukarı bağlantı simülasyon modeli

Mevcut IS-95 CDMA sistemi tabanlı ve ITU tarafından tanımlanan cdma2000 3G standardı olarak kabul edilmiştir. Simülasyonlarda cdma2000 yukarı bağlantı temel bant sinyal modeli olarak radyo konfigürasyonu 1 (RC1) kullanılmıştır. RC1'e göre hareketli kullanıcıda üretilen 1,2288 Mcps veri oranında sinyal baz istasyonuna

$$\mathrm{S(t)} = \sum_{k=1}^{\infty} \mathrm{d(k,t)} \otimes \mathrm{c(k,t)}, \tag{4.48}$$

iletilir. Hareketli kullanıcı tarafından baz istasyonuna iletilen temel bant S(t) sinyali, ile tanımlanır. Burada, k zaman aralığı indeksi, d(k,t), karmaşık yayılımdan önceki dalga formu, c(k,t), karmaşık pseduo noise (PN) yayılım dizisi, $\otimes$ fonksiyonunu,

$$\left(\mathrm{x_1 + jy_1}\right) \otimes \left(\mathrm{x_2 + jy_2}\right) = \mathrm{x_1 x_2 - y_1 y_2 + j\left(x_2 y_1 + x_1 y_2\right)}, \tag{4.49}$$

ile ifade edilir. Burada, d(k,t) sinyali, trafik ve pilot kanal bilgi sembollerinden oluşur. Bu sinyal,

$$\mathrm{d(k,t) = d_I(k,t) + jd_Q(k,t)}, \tag{4.50}$$

ile belirtilmektedir. Burada j eşitlik sanal bölümünü işaret etmektedir. $d_I(k,t)$ ve $d_Q(k,t)$,

$$d_I(k,t) = \sum_{i=1}^{N_C} G_1 \, b((k-1)N_C + i), \tag{4.51}$$

$$d_Q(k,t) = \sum_{i=1}^{N_C} G_2 \, h((k-1)N_C + i) \tag{4.52}$$

ile ifade edilmektedir. Burada, $b(\cdot)$ pilot kanala ait bilgi sembolleri ki bu sembollerin hepsi "1" değerine eşittir, $h(\cdot)$ trafik kanalının veri dizisi ve N_C değeri de her zaman aralığındaki yayılım kodunun uzunluğudur. c(k,t) karmaşık PN yayılım dizisinin dalga formu

$$c(k,t) = \sum_{j=0}^{N_c-1} C^{PN} \, p(t - [(k-1)N_c + j]T_c), \tag{4.53}$$

şeklinde yazılabilir. Burada, T_C çip periyodu ve C^{PN} karmaşık yayılım kodudur.

$$C^{PN} = C_I^{PN} + jC_Q^{PN}, \tag{4.54}$$

burada, C_I^{PN} ve C_Q^{PN} değerleri özdeş olarak dağıtılmıştır ve $\{+1,-1\}$ aralığında eşit olasılıkta rasgele ikili düzende sayı değerleri alırlar.

S(t) sinyali, ortamdaki çoklu yollara maruz kalır ki bu yollar karmaşık yol zayıflaması $\alpha_l = \beta_l e^{j\varphi l}$ ve zaman gecikmesi τ_ℓ ile her çoklu yol sinyali için ayrı ayrı belirlenir. Senaryo gereği hücre ya da sektörde N adet kullanıcının olduğunu varsayalım. O zaman baz istasyonundaki M elemanlı anten dizisinden alınan sinyal,

$$\mathbf{X}(t) = \sum_{\ell=1}^{L} \alpha_\ell \, s(t - \tau_\ell)\mathbf{a}(\theta_\ell) + \mathbf{I}(t) + \mathbf{N}(t), \tag{4.55}$$

ile verilir. Burada, I(t) çoklu erişim ara yüzüdür (Multiple access interference, MAI).

$$\mathbf{I}(t) = \sum_{q=1}^{N-1} \sum_{\ell=1}^{Lq} \alpha_{q,\ell}\, s(t-\tau_{q,\ell})\mathbf{a}(\theta_{q,\ell})\,. \tag{4.56}$$

N(t), Mx1 boyutlu beyaz gürültü vektörü, $a(\theta_\ell)$ Mx1 boyutlu doğrultu vektörüdür ve her çoklu yol için farklıdır, θ_ℓ ise her çoklu yolun geliş açısını verir. Karmaşık temel bant sinyali, karmaşık ağırlık elemanları ile elde edilir.

$$\mathbf{W} = \begin{bmatrix} W_1 & W_1 & .. & .. & W_M \end{bmatrix}^T. \tag{4.57}$$

Ağırlık vektörleri, bazı optimizasyon algoritmaları ile maksimum SINR güç oranını ya da optimum ilgili maliyet fonksiyonunu verecek şekilde elde edilir. Sonuç olarak Şekil 4.1'de görüldüğü gibi anten çıkışındaki sinyal, bu karmaşık ağırlıklarla çarpılır [39].

$$y(t) = \mathbf{W}^H \mathbf{X}(t)\,. \tag{4.58}$$

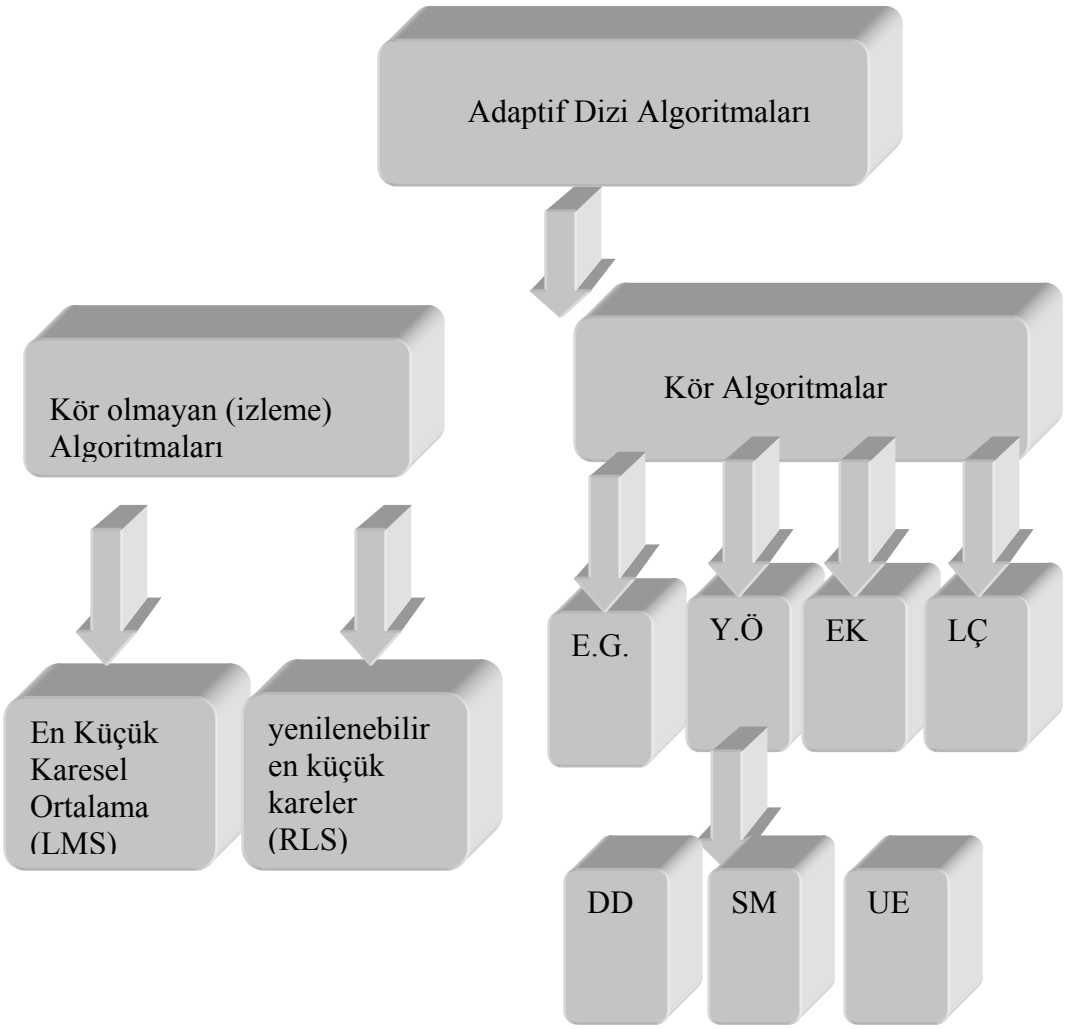
4.4 Adaptif Işın Şekillendirme Algoritmaları

Akıllı antenlerin en önemli kısmını gelişmiş işaret işleme mekanizmaları oluşturur. Çünkü, akıllı antenlerin akıllı olmasının sebebi antenden ziyade çeşitli akıllı anten algoritmaları kullanarak aktif olarak işaret işleme işlemi yapılmasıdır. Bu sebeple algoritma seçimi oldukça önemli bir faktördür. Gezgin istasyon, sürekli hareket halinde olabileceğinden, işaretin geliş açısının tahmini ve de bu tahmin doğrultusunda ışın örüntüsünün oluşturulması seçilen adaptif algoritmalar sayesinde gerçekleştirilir [38].

Eğer, istenen işaretin geliş açısı zamanla değişirse, bir optimizasyon şeması oluşturarak istenen işareti en az hata yaparak en kısa sürede yakalamamız gerekir. Tasarlanan algoritma elektromanyetik ortamın her değişiminde istenen işaret için ayarlama yapmalıdır. Adaptif algoritmalar ilk adımda sabit ışın şekillendirme gibi çalışırlar, daha sonraki adımlarda ise sürekli güncellenen ağırlık vektörleri ile işareti yakalamaya çalışır. Bu ayarlama işlemi özelleştirilmiş bir adaptasyon kriteri tahmini olarak da düşünülebilir.

Dizi ağırlık vektörlerini oluşturmak çok yüksek hesaplama yükü gerektirir. Bu yüzden gerçel zaman içinde optimum çözüm sunabilen teknikler gereklidir.

Adaptif dizi algoritmaları aşağıdaki şekilde olduğu gibi sınıflandırılabilir. Genel olarak bu algoritmalar kör algoritmalar ve kör olmayan algoritmalar olarak ikiye ayrılır [33]. Kör algoritmalar En Az Karesel Ortalama (least mean square, LMS) ve Yenilenebilir En Küçük Kareler (Recursive Least Square, RLS) olarak ikiye ayrılır.



Şekil 4.3 Adaptif dizi algoritmaları

Kör algoritmalar ise Eşlenik Gradyen (Conjugate Gradient), Yenilenen Özellik (Property Restroral), En Küçük Kareler (Least Squares) ve Lagrange Çarpıcı (Lagrange Multiplier) olarak 4' e ayrılır.

Aynı şekilde Yenilen Özellik Algoritması da kendi içinde Direk Karar(Decision Directed), Sabit Modül (Constant Modulus) ve Uzaysal Eşevre (Spectral Coherence) olarak ayrılır.

Yapılan çalışmada En Küçük karesel Ortalama, Sabit Modül ve de önerilen yukarıdaki tabloda da yerleştirilmeyen Uzaysal Kod Korelatör (Space Code Correlator, SCC) gerçeklenmiştir.

4.5 Kör Olmayan Algoritmalar

Bir referans işaret kullanma hesabına dayanır. Bu algoritmalar böylece referans işaret ile istenen işaretin yakınsamasına göre uygun değer geldiğinde sonlandırılır. Şimdi bu algoritmalardan En Küçük Karesel Ortalama ve de Yenilenen En Küçük Kareler algoritmaları incelenecek.

4.5.1 En küçük karesel ortalama (Least mean squares -LMS) algoritması

En az kareler algoritması, gradyan (gradient) temelli bir yaklaşımdır. 1959 yılında ortaya atılan bu algoritma kör olmayan tip algoritmalar grubuna girer. Bu algoritmanın hesaplanabilmesi için pilot işarete ihtiyaç duyulmaktadır. Performans arayüzü dizi ağırlıkları karesel (quadratic) bir fonksiyon olduğundan, j(w) arayüz performansına sahip bu sistem minimum eliptik paraloid fonksiyona sahip olur. Burada performansın en iyi şekilde gerçekleşmesi için minimum karesel hatadan yararlanılır [33,38].
Her iterasyon için güncellenmiş ağırlık vektörleri kullanılır. Hata aşağıda belirtildiği gibi elde edilebilir.

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \mu\nabla_{\omega} \text{ (En Küçük Karesel Hata (MSE))} \qquad (4.59)$$

∇_{ω} En Küçük Karesel Hatanın gradyenidir. Referans (pilot, r(n)) işaret ile dizi çıkışı arasındaki farka dayanan Ortalama Karesel Hata anlamına gelir.

$$MSE(w(t))=E\left[\,|\mathbf{d}(t+1)-\mathbf{w}^{H}(t)\mathbf{x}(t+1)|^{2}\,\right] \tag{4.60}$$

$$=E[|\mathbf{d}(t+1)|^{2}]+\mathbf{w}^{H}(t)\mathbf{R}\mathbf{w}(t)-2\mathbf{w}^{H}(t)E[\mathbf{x}(t+1)\mathbf{r}(t+1)] \tag{4.61}$$

LMS algoritması, içinde MSE'yi en küçük yapan ağırlık değeri çıkış değerinin referans işarete en yakın değeridir. Bu yüzden, araştırmaları bu doğrultuda gerçekleştirebiliriz. MSE karesel forma sahip olduğundan MSE'nin gradyanın negatif doğrultusu içinde taşınan ağırlık vektörleri minimum değere sahip hata değerini oluşturmalıdır [39].

$$\nabla_{\omega} MSE(\mathbf{w}(t)) = 2\mathbf{R}\mathbf{w}(t) - 2E[\mathbf{x}(t+1)d(t+1)] \tag{4.62}$$

$$= 2\mathbf{x}(t+1)\mathbf{x}^{H}(t+1)\mathbf{w}(t) - 2\mathbf{x}(t+1)d(t+1) \tag{4.63}$$

$$= 2\mathbf{x}(t+1)\varepsilon^{*} \tag{4.64}$$

$$\varepsilon = \mathbf{w}^{H}(t)\mathbf{x}(t+1) - d(t+1) \tag{4.65}$$

Buradan aşağıdaki formüllere çıkarım yapılabilir.

$$w(\mathrm{t}+1) = w(\mathrm{t}) + \mu\mathbf{x}(\mathrm{t})\varepsilon^{*}(\mathrm{t}) \tag{4.66}$$

$$\varepsilon(\mathrm{t}) = \mathrm{d}(\mathrm{t}) - \mathrm{y}(\mathrm{t}), \tag{4.67}$$

$$\mathrm{y}(\mathrm{t}) = \mathbf{w}^{\mathrm{H}}(\mathrm{t})\mathbf{x}(\mathrm{t}). \tag{4.68}$$

Burada **w**(t) anten ağırlık vektörü, **x**(t) anten çıkışında kaydedilen sinyal vektörü, ε (t) istenilen kullanıcı pilot sinyali d(t) ile anten çıkışı y(t) arasındaki hata sinyalidir. * ve H işareti sırası ile karmaşık eşlenik ve karmaşık eşlenik transpoze işlemini tanımlamaktadır. μ adım aralığı ise

$$0 < \mu < \frac{1}{10\lambda_{max}}, \tag{4.69}$$

kriterine uygun olarak seçilir ki burada λ_{max} giriş korelasyon matrisinin maksimum özdeğeridir.

Bu eşitliklerde görüldüğü gibi, LMS algoritmasının yakınsaması μ değeri ile orantılıdır. Eğer, adım büyüklüğü çok küçük olursa, yakınsama da küçük olur ve aşırı sönümleme durumu ortaya çıkabilir. Eğer, yakınsama geliş açısının değişiminden daha küçük ise, adaptif dizi değişen işareti izlemek için istenen işareti yeterli hızda yakalayamaz. Eğer, adım büyüklüğü oldukça yüksek ise LMS algoritması ilgili uygun ağırlıkları yakalayamaya bilir. Bu durum underdamped olarak adlandırılır. Yakınsama oldukça hızlı ise, ağırlıklar uygun ağırlıklar gibi salımım yapar. Fakat çözüm tam olarak elde edilemez. Bu yüzden yakınsamayı garanti altına alan bir adım büyüklüğü alınması gerekir.
μ adım aralığı ise

$$0 < \mu < \frac{1}{10\lambda_{max}} \tag{4.70}$$

kriterine uygun olarak seçilir ki burada λ_{max} giriş korelasyon matrisinin maksimum özdeğeridir [39].

4.5.2 Yenilenebilir en küçük kareler

LMS algoritmasının dezavantajlarından biri belirli durumlar altında yakınsama hızı yavaşlar. Örnek vermek istersek, **R** özdeğeri hızı yavaşladığı zamanlarda LMS algoritması doğru sonuçlar vermeyebilir. Bu sebeple RLS algoritması geliştirilmiştir. Burada μ adım aralığı **R** tersi ile yer değiştirir.[40]
Bu algoritma ilk olarak başlangıç ayarlarıyla belirlenir.

$$R^{-1}(0) = \frac{1}{\delta} I, \delta > 0 \tag{4.7}$$

Bu ağırlıklar daha sonra güncellenir,

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \mathbf{R}^{-1}(t+1)x(t+1)\varepsilon^* \tag{4.72}$$

Burada korelasyon matrisinin tersinin güncellenmesi,

$$\mathbf{R}^{-1}(t+1) = \frac{1}{\xi}\left[\mathbf{R}^{-1}(t) - \frac{\mathbf{R}^{-1}(t)\mathbf{x}(t+1)\mathbf{x}^H(t+1)\mathbf{R}^{-1}(t)}{\xi + \mathbf{x}^H(t+1)\mathbf{R}^{-1}(t)\mathbf{x}(t+1)}\right] \tag{4.73}$$

4.6 Kör Algoritmalar

Kablosuz hücreler ve PCS sistemlerin gelişmesiyle değişen ve de bilinmeyen ortam koşulları nedeniyle kör algoritmalar önem kazanmıştır. Bu tip algoritmalar kalibrasyon dizisi gerektirmez. Bazı Yenilenen Özellik gibi algoritmalar, referans işareti ima eden yapıya sahiptir. İstenen işaretin özelliklerini tatmin edici olarak bilirler. Öyle ki En Küçük Kareler gibi algoritmalar istenen işaretin sayısal olarak geçici karakteristiklerinin kullanım temeline dayanır. dizi çıkışı ve iletilen işareti belirlemek için sayısal veriler kullanılır [34].

4.6.1 En küçük kareler (Least Squares)

Standart dizi modeli kullanarak, dizi çıkışında alınan işareti yazabiliriz.

$$\mathbf{X}(t)=\mathbf{A}\mathbf{S}(t)+\mathbf{N}(t) \tag{4.74}$$

Burada,

$$\mathbf{X}(t) = [x_1(t) \quad x_2(t) \quad \ldots x_M(t)]^T \tag{4.75}$$

$\mathbf{S}(t) = [s_1(t) \quad s_2(t) \quad \ldots..s_M(t)]^T$ işaret vektörünü gösterir. (4.76)

$\mathbf{N}(t)= [n_1(t) \quad n_2(t) \quad \ldots..n_M(t)]^T$ gürültü vektörünü gösterir. (4.77)

LS algoritması MLkriterini minimize eder.

$$\min_{s\in\omega} \|\mathbf{X} - \mathbf{AS}\| \tag{4.78}$$

4.6.2 Sabit modül (Constant Modulus) algoritması

Bazı adaptif ışın şekillendirme algoritmaları, pilot (referans) işaret ve çıkış dizisi arasındaki hatanın en aza indirilmesi esasına dayanır. Daha önce değindiğimiz LMS algoritması da bu özelliğe sahip bir algoritmadır. Bu referans işaret bir eğitim dizisidir. Ve de istenen işaret gelen işaretin doğal öncelik bilgisine dayanır. Bu algoritmada bir pilot işaret bilgisi mevcut değildir. Kör (blind) tahmin gelen işareti tam elde edebilmek amacıyla kullanılır. Bazı kablosuz haberleşme sistemleri ve radar sistemleri frekans veya faz modüleli işaretlere sahiptirler. Faz ve frekans modülasyonlu sistemler, FM, PSK, FSK, QAM gibi modulasyonlar kullanır. Bu durumda işaretin genliği idealde sabit olmalıdır. Bu yüzden, işaretin zarf bilgisi istenen işaretin bulunması için yeterli olacaktır. Bu yüzden kör olarak adlandırılır. Bu algoritma sabit değerlikli işaret veya sabit modül algoritma olarak adlandırılır. Fakat bu durumda, sönümleme kanalı içinde çoklu yol terimi ortaya çıkar. Alınan işaret bütün çoklu yol terimlerini bileşimidir. Böylece, kanal işaret büyüklüğü üzerinde genlik değişimi oluşturur. Frekans seçici kanallar işaretin sabit modüleli parçalanması ile bulunur. Eğer biz gelen işaretin sabit modüleli bir işaret olduğunu biliyorsak, algoritmaları orijinal işaretin genliğini dengeleyecek şekilde tasarlarız. Domaniq and Godard iki boyutlu veri haberleşme sistemi içinde kullanılan bir kör algoritma ailesini oluşturarak sabit modüleli işaretten yararlandı. Bu algoritma, özellikle faz modüleli işaret şekli uygulamıştır. Bu algoritma, genel olarak çok avantaja sahip olmasının yanında yakınsamanın belirli durumlarda sonsuza gitmesi ve de bazı durumlarda istenen kullanıcıdan ziyade parazit işaretin yakalanması gibi dezavantajlara sahiptir [41].

Bu algoritma çıkış işaretinin özelliklerini kullanarak hata işaretinin elde edilmesini sağlar.CDMA sistemlerde güç kontrolü kullanıldığından dolayı her kullanıcının işaret gücü servis kriterinin kalitesi için ayarlanır. Bu yüzden algoritma bu algoritma CDMA sstemler için önerilmez. Yapılan çalışmada CM algoritmasının da

performansı değerlendirilmiş, fakat daha iyi sonuçlar veren SCC algoritması önerilmiştir.

$$y(t) = \mathbf{w}(t)\mathbf{x}(t), \tag{4.79}$$

$$\varepsilon(t) = 4y(t)(\left|y(t)^2\right| - 1), \tag{4.80}$$

$$\mathbf{w}(t+1) = \psi\,\mathbf{w}(t) - \mu\,\mathbf{x}(t)\varepsilon^*(t), \tag{4.81}$$

burada, ψ bire yakın skaler bir sayı, μ adım aralığı parametresidir. Bununla beraber literatürde çeşitli hata tespit algoritmaları bulunmaktadır.

$$\varepsilon(t) = \frac{y(t)}{\left|y(t)\right|}\,\mathrm{sgn}(\left|y(t)\right| - 1), \tag{4.82}$$

$$\varepsilon(t) = 2y(t)\,\mathrm{sgn}(\left|y(t)\right|^2 - 1), \tag{4.83}$$

$$\varepsilon(t) = 2\left(y(t) - \frac{y(t)}{\left|y(t)\right|} \right), \tag{4.84}$$

4.6.3 Uzay kod korelatör (Space Code Correlator –SCC) Algoritması

CDMA kablosuz sistemlerde bütün kullanıcılar aynı zamanda aynı frekansı kullanır. Temel istasyon, ayrık olarak her yayılma dizisini çözerek istenen işaretin herbirini ayırır. Bu farklı ışın şekillendirme teknikleri CDMA sistemleri için aşağıdaki sebeplerden dolayı uygun değildir.

Öncelikle CDMA kablosuz sistemler içinde bütün kullanıcılar ve bu kullanıcılarla beraber yardımcı kanal sayıları kolaylıkla anten sayısını geçebilir. Dahası haberleşme ortamı çoklu yol ortamı olduğundan dolayı, her iletim yolu direk, yansımalı ve de saçılmalı yol içerir.

Yayılma ortamı bu dezavantajlara sahip olduğundan dolayı adaptif ışın şekillendirme algoritması olarak Uzay Kod Korelatör (Space Code Correlator -SCC) algoritması

önerilebilir. Öncelikle bu algoritma Kod Korelatör ve de Uzaysal Korelatör olarak iki kısımdan oluşur. Kod korelatör kısmında istenilen kullanıcı anten girişine gelen işaretin istenilen kullanıcı kodu ile korelasyonu gerçekleştirilir. Uzaysal korelatör kısmı ise hangi açılarda işaretin maksimum olduğunun belirlenmesini sağlar. [36]

$b_i(l)$ sıfır ortalamalı i. kullanıcı varyans birimi ile birlikte kimliksel bit dizisi olarak yayılır. $c_i(l;m)$ belirlersek *m=0...L-1* kullanıcı için *l.* sembolün yayılım kodu L-çip olur.

$$c_i(l;t) = \sum_{m=0}^{L-1} c_i(l;m) p\left(t - m\frac{T_w}{L}\right) \tag{4.85}$$

Burada p(t) darbe biçim süzgeci, T_c çip periyodu ve (T_w/L) işlem kazancı üzerinden sembol oranı olarak isimlendirilir.
Temel band işareti

$$s_i(t) = \sum_{l=-\infty}^{\infty} b_i(l) c_i(l;t) \tag{4.86}$$

Temel band işareti cdma2000 reverse link radyo konfigürasyonu kullanılarak üretilir. Temel istasyonda M elemanlı anten dizisi olarak alınan işaret,

$$\mathbf{X}(t) = \sum_{i=1}^{N} \sum_{f=1}^{F_i} \alpha_{i,f} s_i(t - \tau_{i,f}) \mathbf{a}(\theta_{i,f}) + \mathbf{n}(t) \tag{4.87}$$

f çokluyol indeksini, F çokluyol sayısını bildirir.

$$\mathbf{X}(t) = \underbrace{\sum_{f=1}^{F_1} \alpha_{1,f} s_1(t - \tau_{1,f}) \mathbf{a}(\theta_{1,f})}_{istenen} + \underbrace{\sum_{i=2}^{N} \sum_{f=1}^{F_i} \alpha_{i,f} s_i(t - \tau_{i,f}) \mathbf{a}(\theta_{i,f})}_{girişir} + \mathbf{n}(t) \tag{4.88}$$

Bu çoklu yol kanalı $\alpha_{i,f} = \beta_{i,f} e^{j\phi_{i,f}}$ karmaşık çokluyol zayıflamasına sebep olur. $\tau_{i,f}$ zaman gecikmesini verir. Kod korelatör kısmında, ilk olarak alınan işaret çoklu yol elemanları istenen kullanıcı kodu C tarafından despread yapılır. Burada C,

$$C_l(t) = C(t - \tau_l) \tag{4.89}$$

$$\mathbf{Z}^{(p)}(t) = \frac{1}{T_c} \int_{(l-1)T_w}^{lT_w} \mathbf{X}(t) c_1^*(t - \tau_1) dt \tag{4.90}$$

Burada, p 1 den M e değişen yol saysısını verir. Her bir çoklu yol bileşeni için despread edilmiş sinyal

$$Z_l^k(t) = \frac{1}{N_c} \sum_{j=0}^{N_c-1} \bar{x}_l(t - [(k-1)N_c + j]T_c) C_l^*(t - [(k-1)N_c + j]T_c) \tag{4.91}$$

formülü ile hesaplanır. Burada T_c çip periyodunu, N_c çip sayısını ve k sembol aralığını temsil etmektedir.

Her yol için kod korelatör işleminden sonra uzaysal korelasyon işlemi $\mathbf{Z}^{(p)}(t)$ ve dizi cevap vektörü arasında gerçekleştirilir. Bu amaç için, $\Delta\theta°$ tarafından oluşturulan 0° ile 180 arasında K DOA lara bölünür. θ_K=180°/*K olur.* Bu yüzden reverse link arama tablosu içinde K yönlendirme vektörlerinin toplanmasını ihtiyaç duyulur. Her çokluyol için uzaysal krelatör çıkışı maksimum DOA bulmak için çalışır. Kod korelatör çıkışından gelen işaret ile verilen açı aralığında K adet açıdan elde edilen Mx1'lik yönlendirme vektörü iç çarpımının genliği uazysal korelatör ile elde edilir.

SCC algoritmasında, despread işlemi için gereken her bir çoklu yol zaman gecikmesinin (τ_l), başka bir zaman gecikmesi tarayıcı (time delay search) algoritması tarafından hesaplandığı varsayılmaktadır. Böylece, zamansal bölgede yayılmış olan sinyaldeki örnekleme sayısı yayılım faktörü oranında küçültülür.

$$\hat{\theta}_l = \arg\max_{\theta} \left\{ \left| \bar{a}(\theta_l)^H \bar{Z}_l(t) \right| \right\}^2 \tag{4.92}$$

Burada $a(\theta_i)$ 0-180° arasında belirlenen çözünürlükteki sinyal kaynaklarının yön vektörüdür.

$$\bar{a}(\theta_i) = \left[1 \quad e^{j 2\pi d \sin\theta_i / \lambda} \quad . \quad . \quad e^{j 2\pi (M-1) d \sin\theta_i / \lambda} \right]_0^{180}, \; 0 \le \theta_i \le 180 \tag{4.93}$$

Bu işlem sonucunda, K adet açıdan hangisi maksimum değeri veriyorsa o açı çoklu yol sinyalinin geliş açısıdır. Aynı işlem, L adet çoklu yol sinyali için eşzamanlı olarak hesaplanır. Sonuçta, ışın şekillendirici için kullanılacak olan ağırlık vektörü, çoklu yol açılarının ortalamasından elde edilen açıdan oluşturulan yön vektörüdür.

Tahmin edilen DOA birkaç sembol periyodu boyunca yavaş yavaş kullanıcı pozisyonu değiştikçe aşağı bağlantılı ışın şekillendirme ve yukarı bağlantılı ışın şekillendirme için kullanılır. Küçük çokluyol korelasyon seviyesi ve küçük DOA çözünürlüğü için algoritmanın DOA istenen kullanıcıyı bulması gerekir.

SCC algoritmasının bir avantajı ağırlık vektörü hesaplama zamanının değişik propagasyon koşullarından ve anten topolojilerinden bağımsız olmasıdır, fakat hesaplama zamanı çoklu yol sayısı (L) ve uzaysal tarama aralığındaki açı sayısına (K) bağlı olarak değişir [36].

Tablo 4.1' de algoritmaların hesaplama karmaşıklığı gösterilmiştir.

Tablo 4.1. Algoritmaların hesaplama karmaşıklıkları.

	Toplama	**Çarpma**
LMS	(8m+2)*i	(11m+1)*i
CM	(8m+5)*i	(13m+10)*i
SCC	6036m+4320	6012m+4320

Not: m=anten sayısı, i=algoritmaların iterasyon sayısı.

BÖLÜM 5. AKILLI ANTEN ALGORİTMALARININ FPGA TABANLI GERÇEKLENMESİ

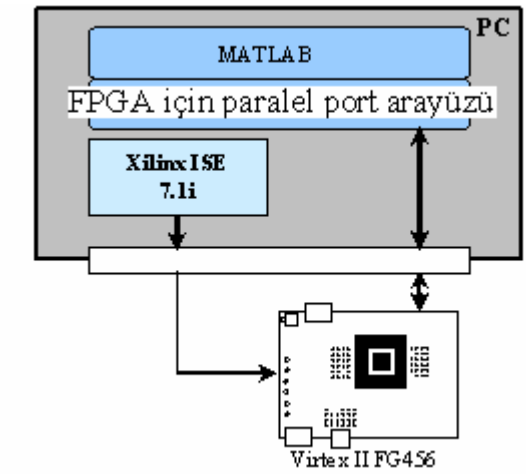
5.1. Giriş

Bu bölümde, LMS, CM ve SCC algoritmalarının FPGA üzerinde gerçeklenmesi üzrerinde duruldu. Uygulamada, özellikle hata, mimari karmaşıklığı ve de hesaplama zamanı kavramları incelendi. Burada, cdma2000 reverse link işaret modeli uygulandı. Çeşitli ortam durumlarında elde edilen işaretler için ULA anten topolojisi kullanılmıştır. Akıllı anten algoritmalarının FPGA üzerında gerçeklenmesi için, VHDL dilinde kullanabilen IEEE 754 tek duyarlılıklı kayan noktalı sayı formatı kullanıldı.[42] Bu format ile sayılar 32 bitlik tek duyarlılıkla veya 64 bitlik çift duyarlılıkla ifade edilebilir. Yapılan çalışmalarda, kullanılan bit sayısı arttıkça sistemin hassasiyeti artacak fakat bu da sistemin karmaşıklığını arttıracaktır. Yapılan çalışmada, değişken (generic) kullanılarak algoritmaların farklı hassasiyetlerde çalışması sağlanmıştır. Özellikle 32 bitlik mimari gerçeklenmeye çalışılmış, fakat önerilen SCC algoritması çok yüksek işlem yüküne sahip olduğundan dolayı yalnızca 16 bitlik kayan noktalı sayı kullanılarak gerçeklenebilmiştir. Fakat kullanılan bit sayısı azaldıkça sistemin hassasiyeti azalacak bu da algoritmaların gelen işaret açısını çok yüksek bir hata ile yakalaması anlamına gelecektir. Örneğin, 8 bit değerin kullanılması işaretin çok farklı bir değerde yakalanması anlamına gelir. IEEE 754 tek duyarlılıklı kayan noktalı sayı formatı sistem karmaşasını çok arttrıracağı için yapılacak en basit toplama ve çarpma işlemleri için bile modüller kullanılacaktır. Bu modüller ileri de detaylı bir şekilde anlatılacaktır.

Antenden alındığı varsayılan giriş işaretinin üretilmesi, üretilen sinyallerin tamamının IEEE 754 kayan noktalı sayı formatına dönüştürülmesi ve dönüştürülen sayıların FPGA kartına yüklenmesi işlemi MATLAB yazılımı ile gerçekleştirilmiştir.

Karta yükleme işlemi için MATLAB da yazılan ve parelel portu kullanan bir fonksiyon kullanılarak FPGA kartının harici pinlerinden yükleme işlemi yapılmıştır.

Yazılan algoritmaların FPGA kartına yüklenmesi işlemi ise Xilinx ISE 7.1i yazılımı vasıtasıyla PC den gerçekleştirilmiştir [43].



Şekil 5.1 Matlab ve FPGA Bağlantısı

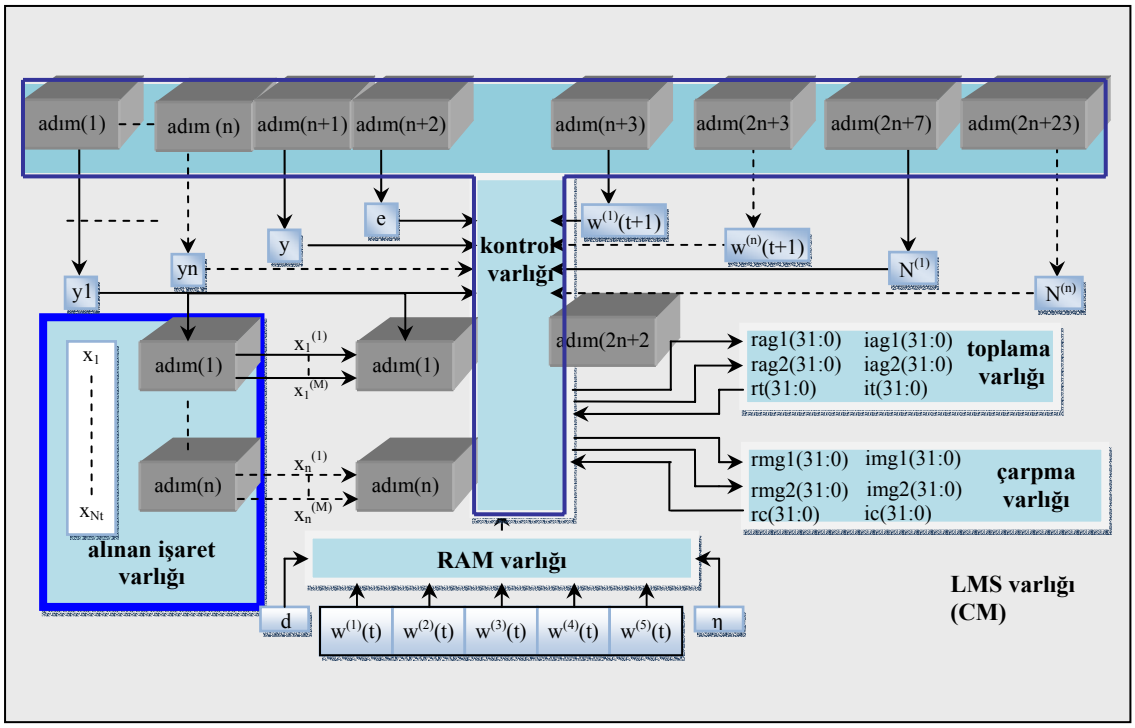
Daha önce de ifade edildiği gibi, LMS, CM ve SCC algoritmalarının FPGA de gerçekleştirilmesini incelenecek. Burada ilk olarak birbirine çok benzeyen yalnızca hatanın hesaplanmasında farklılık gösteren ve de bu özelliklerine göre kör ve kör olmayan tip algoritmalar olarak adlandırılacak LMS ve CM algoritması değerlendirilecek.

5.2. LMS ve CM Algoritmalarının FPGA Tabanlı Gerçeklenmesi

Bir önceki bölümde anlatıldığı gibi LMS kör olmayan tip bir algoritma olup, CM kör algoritmadır. LMS algoritmasında hata bulmak için pilot (referans) işaret kullanılırken CM algoritması hatanın çıkış değerine göre hesaplanması sonucu referans işarete gerek duymaz. Burada LMS algoritması ile CM algoritması arasındaki fark hatanın hesaplanması olarak karşımıza çıkar.

Esas işlemin yapıldığı yer kontrol varlığı olarak adlandırılan varlık olup, algoritmanın temel prensibi algoritma çıkışında elde edilen ağırlık vektörlerinin sürekli değiştiği varsayılan anten giriş değerleri ile işlenmesidir. Şimdi, yapılan işlemler modüller halinde incelenecek.

Şekil 5.2' de gözüktüğü gibi algoritma temel olarak 5 varlıktan oluşur.



Şekil 5.2 LMS ve CM algoritmalarının FPGA'de gerçeklenmesi

Bu varlıklar alınan işaret varlığı, RAM varlığı, kontrol varlığı, toplama varlığı, çarpma varlığıdır. Genel olarak varlıkların işlevini verdikten sonra algoritmanın çalışma mantığı alt kısımda anlatılacak.

Alınan işaret varlığı: Bu varlık bir tampon görevi görür. Antenden alınan işaretlerin depolandığı yerdir. Alınan işaret değerleri karmaşık sayılardan oluşan IEEE 754 kayan noktalı sayı formatında olup, her iterasyon için alınan işaret değerleri değişir. Burada M sayısı anten sayısını gösterirken, n her iterasyon için alınan farklı anten işaret değerlerini gösterir. Yapmış olduğumuz çalışmada 5 anten kullanılmıştır. Antenden alındığı varsayılan işaret sayısı iterasyon sayısıyla birlikte ortam koşulları ile değişebilir. Çünkü belirlemiş olduğumuz kriter değerine göre iterasyon devam edeceğinden dolayı daha önceden iterasyon sayısı bilmemiz mümkün değildir. Bu da antenden alınan işaretlerin kaçının kullanılacağının önceden bilinmemesi anlamına gelir. Yapılan uygulamada LMS ve CM algoritması için aynı ortam koşularında üretilen cdma2000 formatında işaret kullanılmıştır. İki iterasyon birbirinden farklı sürmüştür.

RAM varlığı: Bu varlıkta algoritmanın başlatılabilmesi için gerekli olan antenden alındığı varsıyan gelen işaret değerleri, başlangıç ağırlık değerleri ve adım aralığı aralığı tutulur. Sonraki iterasyonlar için yenilenen ağırlık vektörleri değerleri alındığından, her iterasyon için programın doğru bir şekilde çalışabilmesi için farklı ağırlık değerleri RAM varlığından çağrılır.

Toplama varlığı: IEEE 754 kayan noktalı sayı formatı kullanıldığından basit bir toplama işleminin dahi FPGA üzerinde gerçekleştirilmesi oldukça zordur. Bu yüzden özel bir algoritma ile elde edilen toplama varlığı bu formatta işlemlerin yapılması için kullanılır. Bu varlık IEEE 754 kayan noktalı sayıların toplanması ve dolaylı olarak çıkarılması işlemlerini gerçekleştirir. Bu varlık karmaşık toplama işlemini başarmak için 4 defa toplama fonksiyonunu çağırır.

Çarpma varlığı: Bu varlık da toplama varlğında olduğu gibi farklı bir algoritma kullanarak IEEE 754 kayan noktalı sayıların çarpma işlemini gerçekleştirir. İşlem karmaşıklığı toplama işlemine göre daha azdır. Karmaşık çarpma işlemi için 2 defa toplama 4 defa ise çarpma fonksiyonu çağrılır.

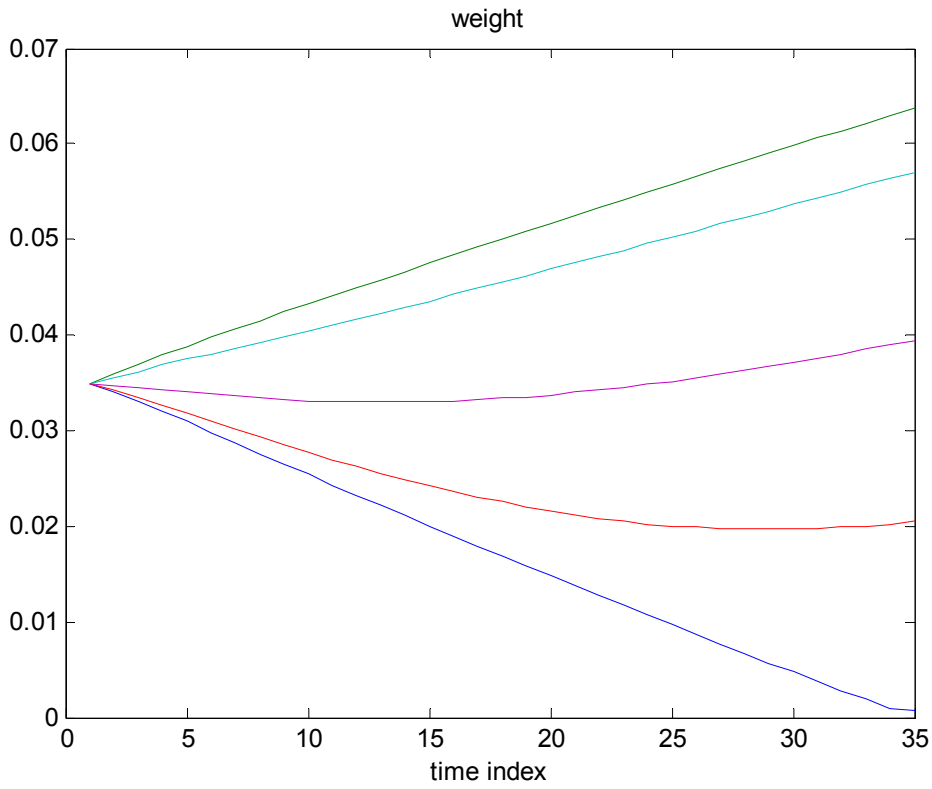
Kontrol varlığı: Bu varlık esas işlemlerin yapıldığı varlıktır. Genel olarak işlemleri şu şekilde özetleyebiliriz.

LMS ve CM algoritmaları için, cdma2000 formatında işaret üretilmiştir. Cdma2000 formatında işaret için matlab programı kullanılmıştır. Burada istenen işaret ve girişim işaretinin değerleri farklı zayıflama krieterlerine bağlı olarak üretilmiştir. Gürültü işareti ise SNR=20dB olarak üretilmiştir.

Öncelikle, ilk iterasyonda antenlere ait olan işaret değerleri(X_1, X_2, X_3, X_4, X_5) ile RAM varlığında bulunan ilk ağırlık değerleri(W_1, W_2, W_3, W_4, W_5) çarpılır. Burada, başlangıç ağırlık değeri 5X1' lik vektördür. Yine, her iterasyon için antenden alınacak giriş değerleri de 5X1' lik vektördür. Ağırlık değerlerinin karmaşık eşleneği alınarak 1X5' lik başlangıç ağırlık değerleri ile 5X1' lik anten giriş değerlerinin çarpımı sonucu 1X1' lik karmaşık bir skaler değer olarak anten dizisi çıkışı elde edilmiş olunur.

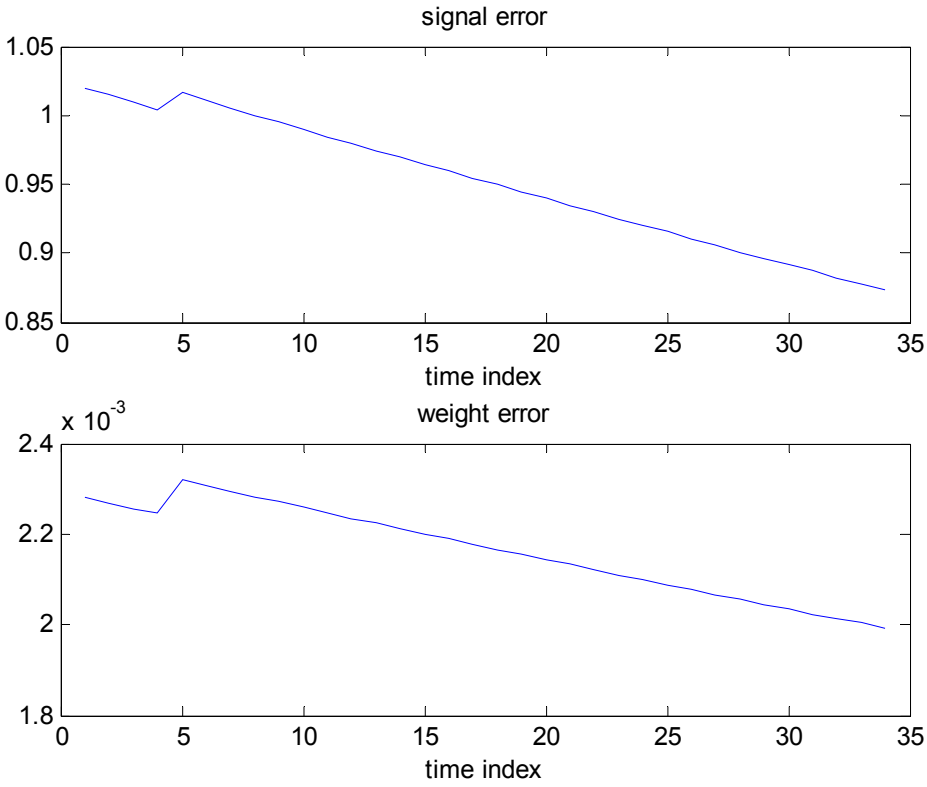
Bir sonraki aşamada, hata değerinin hesaplanması gelir. Bu nokta LMS ve CM algoritmaları için farklılık oluşturur. LMS algoritması hatanın hesaplanması için referans işaret (d) kullanır. Hata referans işaret değerinden elde edilen çıkış değerinin çıkarılması ile bulunur (d-y). CM algoritmasında ise referans işareti kullanılmaz. Burada, pilot işaret olarak tüm iterasyon değerlerinde '-1' değeri kullanılır. Yine hata değeri skaler bir değer olarak elde edilir. Bir sonraki kısım ise güncellenmiş ağırlık değerlerinin bulunmasıdır. Burada, güncellenmiş ağırlık değerlerinin bulunması için adım aralığı μ ifadesi kullanılır ($\mathbf{w}(t+1) = \mathbf{w}(t) + \mu\mathbf{x}(t)e^{*}(t)$). Adım aralığı ise, $0 < \mu < \frac{1}{10\lambda_{max}}$ kriterine uygun seçilir ki burada λ_{max} giriş korelasyon matrisinin maksimum değeridir.

Dördüncü aşama ise daha önceden belirlemiş olduğumuz kriter ile durdurma değeri karşılaştırılır. Burada durdurma değeri ise şu şekilde elde edilir. Güncellenen ağırlık vektörlerinden önceki ağırlık vektörü değerleri çıkarılır. Bu fark değerler arasında norm işlemleri yapılarak kriter değer ile karşılaştırma yapılacak değer elde edilir. Bu noktada elde edilen değer eğer kriter değerden büyükse iterasyon devam ettirilir. Eğer elde edilen değer küçük ise iterasyon sonlandırılır. Eğer iterasyon devam ediyorsa, bir sonraki aşama için alınan işaret varlığından alınan işaret değerleri yenilenir. Ağrlık değerleri için de güncellenmiş ağırlık değerleri kullanılır. Kriter değerden küçük olana kadar işlem devam ettirilir. Özellikle burada norm alma işlemi üzerinde durmak gerekir. Çünkü norm alma işlemi karekök alma işlemi gerektirdiğinden bizim için önemli bir iş yükü gerektirir. Bu yüzden karşılaştırma işlemini normu bulunanan ifadenin karesini kullanarak kolaylıkla gerçekleştirebiliriz. Böylece işlem yükünü önemli ölçüde azaltmış oluruz. Burada karesi alınmış ifade çok küçük bir değer olması halinde FPGA ve 32 bitlik kayan noktalı sayı formatından kaynaklanan hatalarla karşılaşılabilir. Fakat bu yapılan çalışmada çok önemli hatalara sebep olmamıştır. Şimdi LMS algoritması için elde edilen sonuçlar incelenelecek.



Şekil 5.3 LMS algoritması için ağırlık değerlerinin genliği

Burada, durdurma değeri ile kriter değer her iterasyonda karşılaştırılması sonucu 35. iterasyona kadar devam eden ağırlık değerlerinin genliği çizilmiştir. Matlab ile FPGA sonuçları karşılaştırıldığında ağırlıkların genlik değerlerinin birbirine çok yakın olduğu gözlemlenmiştir. Şekil 5.3' de LMS algoritmasının ağırlık değerlerinin genliği gösterilmiştir. Dikkatli bir şekilde incelendiğinde, ağırlık değerlerinin salınım yaptığı ve de 35. iterasyonda durağan hale geldiği fark edilir.



Şekil 5.4 LMS a. Hata işaretinin genliği, b.Durdurma değeri

Şekil 5.4.a' de ilk olarak hatanın genliğinin her iterasyondaki değeri verilmiştir. Burada, her adımda '-1' değerine sahip pilot işaretten çıkış dizisi çıkarılır. İkinci şekilde ise iteratif algoritmalar için önemli olan durdurma değeri bulunur. LMS algoritmasında iterasyonu durduracak değer olarak 0.0029 değeri yani yaklaşık %0.2 değeri daha iyi sonuçlar vermesi açısından özellikle seçilmiştir. Burada FPGA' de bulmuş olduğumuz değerlerin karekökü alınarak çizim gerçekleşmiştir. Şekil 5.4.b' de ise hata değerinin zamana göre değişimi incelenmiştir. Bu şekilde hatanın 35. iterasyonda çok küçük bir değere ulaştığı kolaylıkla fark edilebilir. Kriter değerden küçük olduğu için algoritma sonlandırılmıştır.

Süre hakkında genel bilgi verecek olursak, LMS algoritması 35. iterasyona kadar devam etmektedir. FPGA' de durdurma değeri ile kriter değer arasındaki karşılaştırmaya bağlı olarak iterasyon değeri antenden farklı değerler alınarak veya ortamın özellikleri değiştirilerek değişebilir. Burada verilen değerlere göre 35 defa algoritma döner.

Her iterasyon FPGA'de 18400 picosaniye= 0.0184 mikro saniye sürmektedir. Genel olarak 35x0.0184=0.6072 mikro saniye işlem sürer. Oldukça iyi sonuç verdiği görülmektedir. Daha sonra detaylı olarak süre tablo halinde verilecektir.

LMS algoritmasının son kısmında ise son güncellenen ağırlık vekörü değerlerinin matlab sonuçları ile FPGA sonuçları karşılaştıılmıştır. Önce matlabda elde edilen değerlerin 35. İterasyon değerlerine bakalım.

w
ans =

-0.0002 + 0.0006i 1.ağırlık değeri reel+sanal
0.0630 + 0.0095i 2.ağırlık değeri reel+ sanal
0.0061 - 0.0196i 3.ağırlık değeri reel+ sanal
0.0511 + 0.0255i 4.ağırlık değeri reel+ sanal
0.0230 - 0.0321i 5.ağırlık değeri reel+ sanal

Burada 32 derecede olması istenen işaret yaklaşık 34 derece olarak yakalanabilmektedir.
w=
-0.0008 + 0.0006i 1.ağırlık değeri reel+ sanal
0.0623 + 0.0092i 2.ağırlık değeri reel+ sanal
0.0069 - 0.0191i 3.ağırlık değeri reel+ sanal
0.0507 + 0.0248i 4.ağırlık değeri reel+ sanal
0.0233 - 0.0312i 5.ağırlık değeri reel+ sanal

Son olarak LMS algoritmasının kullandığı kapasite hakkında bilgi verilecek. Elimizde iki adet kit bulunduğundan dolayı bu kitlerin kullanmış olduğu kapasiteler tablo halinde verilecektir. Bu kitler Xilinx ailesine ait xc2v3000 ve xc4vlx60 kitleridir. Özellikle xc4vlx60 kiti oldukça yüksek kapasite değerlerine sahiptir.

Tablo 5.1: LMS algoritmasının (seri) Virtex2 ve Virtex4 için sentez sonuçları

Kullanılan FPGA	Virtex2(xc2v3000)	Virtex4(xc4vlx60)
Algoritma	**LMS**	
Dilim sayısı	14,336'dan 7,019'sı %48	26,654'dan 6762'i %25
Flip-Flop sayısı	28,672'den 3921'si %13	27,392'den 3921'si %7
4 girişli LUT sayısı	28,672'den 9,648'sı %33	53,248'den 9203'i %17
Giriş çıkış sayısı	484'dan 321'i %66	448'dan 321'i %71
GCLK sayısı	16'dan 1'i %6	32'dan 1'i %3

NOT: Ek olarak virtex4 DSP48 yapısına sahip olup, kaynak kullanımı 64'ün 16 sı olup %25 kaynak kullanımı getirir.

Bu gerçekleştirme sonucunda LMS algoritmasının kaynak kullanımı Tablo 5.1' deki tabloda verilmiştir. Burada bazı faklı kısımlar bulunmasına rağmen genel olarak karşılaştırma yapılmıştır.

Yukarıda 32 bitlik kayan noktalı sayı formatı kullanılarak sonuçlar elde edilmiş ve de matlab sonuçlarına yakın değerler elde edilmiştir. Alternatif olarak daha düşük hassasiyeye sahip 16 bitlik sayı kullanılarak da işlemler yapılabilir. Özellikle, 16 bitlik mimari kullanırsak, işlemlerimizi paralel olarak gerçekleştirebiliriz. FPGA' in temel yapısı paralel çalışma mimarisi olduğundan dolayı bu işlem süre açısından oldukça iyi sonuçlar ortaya koyacaktır.

Paralel çalışma ile LMS algoritmasının bir iterasyonu 1800ps=0.0018 mikro saniye sürer. Toplam süre göz önüne alındığında ise bu süre 35x0.0018=0.059 mikro saniye olacaktır. Yukarıdaki işlemler ile karşılaştırdığımızda seri çalışmaya göre paralel çalışmanın 10 kat daha hızlı gerçekleştiği kolaylıkla fark edilebilir.

Fakat, paralel çalışmada toplama ve çarpma modüllerinde 16 bitlik işlem yapıldığından bazı problemler çıkmakta ve de bu modüllere çok küçük değerler geldiğinde istemediğimiz çok büyük değerler elde edilmektedir. Bu sebeple, hatalı sonuçlar verilebileceği için bir de 32 bitlik sayılarla işlem yapmak daha hassas sonuçlar verdiğinden 32 bitlik LMS algoritması üzerinde durulmuştur.

Fakat, problem çözülmesi halinde kapasite sonuçları aşağıdaki gibi olacaktır. Daha önce de ifade edildiği gibi paralel olarak her clock darbersinde işlem yapmak kapasite açısından dezavantaj sağlayacaktır. Bu sebeple 32 bitlik paralel mimari hem virtex2 hemde virtex4'de sığmamıştır.

16 bitlik paralel mimari ve 32 bitlik seri olarak sayaç kullanılarak gerçekleştirilen mimari farklı amaçlarla kullanılabilir. Örnek vermek gerekirse eğer kullanıcı hem LMS hem de CM algoritmaları FPGA içine sığsın diyorsa, 32 bitlik kapasiteyi az kullanan mimari tercih edilir. Faka,t süre önemli ise 16 bitlik parale mimari ile hızlı bir şekilde algoritmaların bir tanesi FPGA' e sığdırılarak işlem yapılabilir. Hatalı sonuçlar verebileceği muhtemel bu problemlerin çözülmesinde elde edilecek kapasite Şekil 5.2' de gösterildi.

Tablo 5.2: LMS algoritmasının (paralel) Virtex2 ve Virtex4 için sentez sonuçları

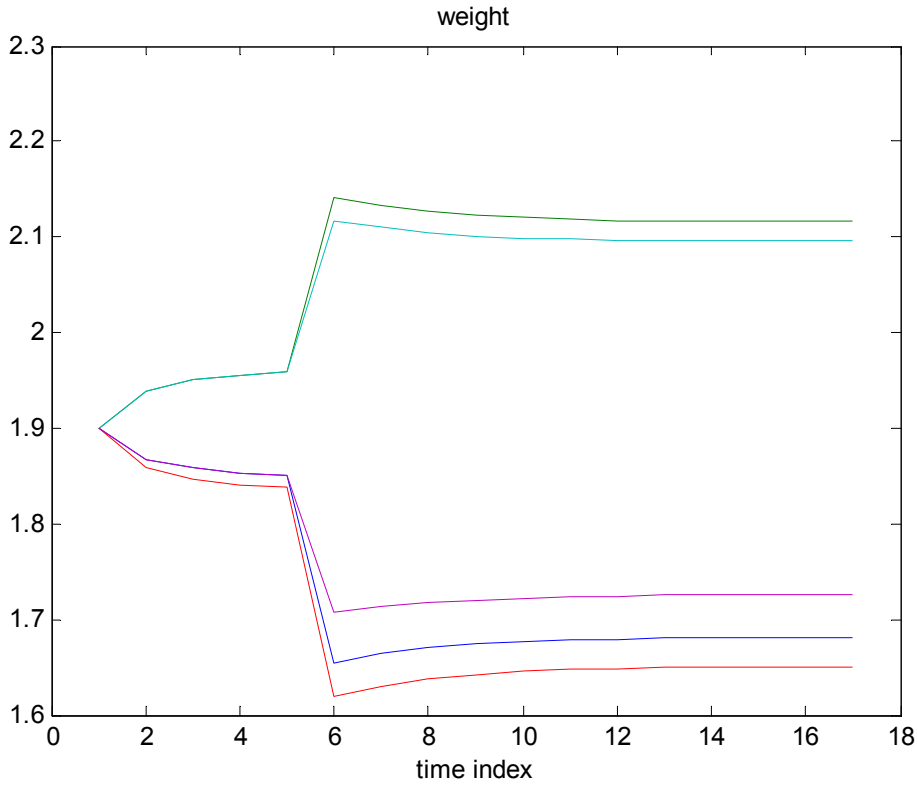
Kullanılan FPGA	**Virtex2(xc2v3000)**	**Virtex4(xc4vlx60)**
Algoritma	**LMS**	
Dilim sayısı	14,336'dan 1,774'sı %53	26,654'dan 7,775'i %29
Flip-Flop sayısı	28,672'den 3,921'si %13	27,392'den 1,709'si %3
4 girişli LUT sayısı	28,672'den 14596'sı %50	53,248'den 14,819'i %27
Giriş çıkış sayısı	484'dan 161'i %33	448'dan 161'i %35
GCLK sayısı	16'dan 1'i %6	32'dan 1'i %3

NOT: Ek olarak virtex4 DSP48 yapsına sahip olup 64'ün 44 yani %68'lik kaynak kullanımı söz konusudur.

Birçok noktada, LMS algoritması ile CM algoritması aynı gözükür. CM algoritması hata bulma işleminde LMS algoritmasından ayrılır. Daha önce de ifade edildiği gibi

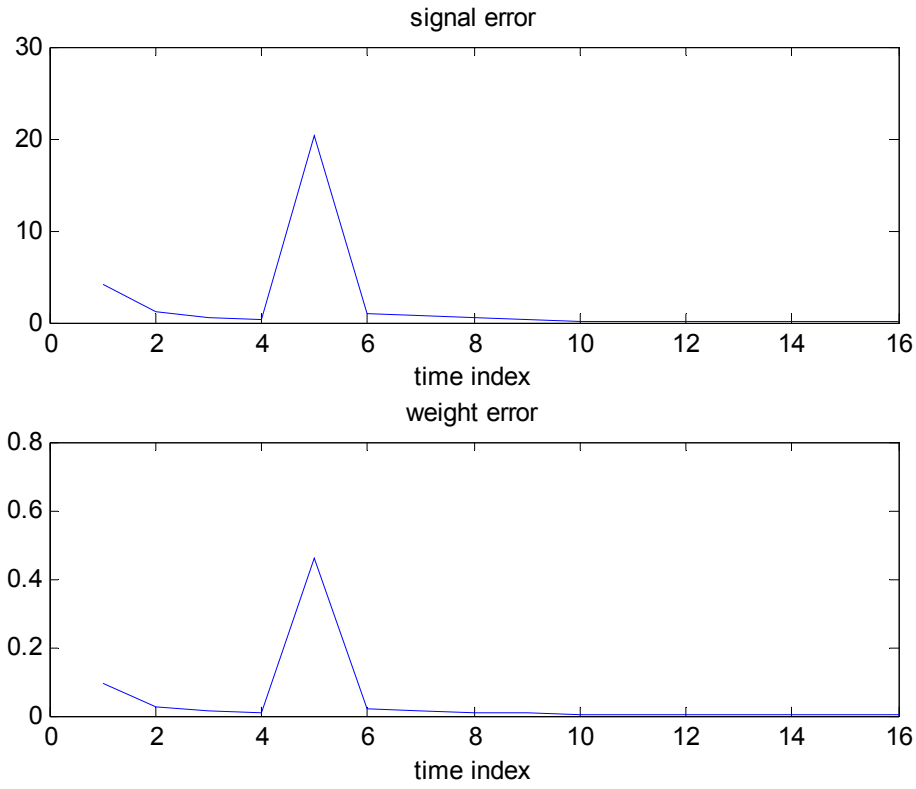
LMS algoritması hatanın bulunması için pilot işaret kullanıyordu. Fakat CM algoritması ağırlık değerleri ile antenden alınan giriş değerlerinin çarpılması sonucu oluşan çıkış dizisinin zarfını kullanır. Bunun için 4. bölümde üzerinde durulduğu gibi farklı hata bulma formülleri kullanılabilir. Fakat genel olarak formüller incelendiğinde norm ifadesinin kullanıldığı görülür. Norm ifadesi karekök alma işlemi gerektirdiğinden dolayı bu formüllerden biri olan ($e(t) = 4y(t)(|y(t)^2| - 1)$) ifadesi tercih edilmiştir. Burada ifadenin karesinin normu olduğu için eşlenik transpoze ile çarpımı doğru ifadeyi verecektir.

Başlangıç ağırlık değerleri LMS algoritmasından farklı olarak alınır. CM algoritması yakınsama problemine sahip bir algoritma olduğundan daha iyi sonuçlar alınabilmesi için başlangıc ağırlık değerleri her anten için '1.9' olarak alınır. Burada başlangıçta çok büyük hata değerleri elde edilir. Fakat itersayon devam ettiği sürece hata değeri giderek azalır. Buna paralel olarak kriter değer ile karşılaştırma işlemi yaptığımız durdurma değeride belirli bir iterasyonda kriter değerden küçük olur ve iterasyon sonlandırılır.



Şekil 5.5 CM algoritması ağırlık değerlerinin genlik değerleri

CM algoritmasının ağırlık değerleri de incelendiği taktirde yine LMS algoritmasında olduğu gibi durağan bir hal aldığı Şekil 5.5' den rahatlıkla fark edilebilir..



Şekil 5.6:CM a. Hata işaretinin genliği, b.Durdurma değeri

Şekil 5.6a'da ilk olarak hatanın genliğinin her iterasyondaki değeri verilmiştir. Burada, hata hesaplaması için $e(t) = 4y(t)(|y(t)^2| - 1)$ formülünden faydalanılır. Şekil 5.6b' de ise ise iteratif algoritmalar için önemli olan durdurma değeri bulunur. LMS algoritmasında iterasyonu durduracak değer olarak 0.0029 değeri yani yaklaşık %0.29 değeri daha iyi sonuçlar vermesi açısından özellikle seçilmiştir. CM algoritmasında ise yakınsama problemi ile karşılaşılabileceğinden dolayı kriter değer oldukça küçük seçilmesinde fayda bulunmaktadır. Bu sebeple, kriter değer 0.00029 yani LMS algoritmasından 10 kat daha hassas alınmıştır.

Süre olarak, CM algoritması incelendiği takdirde, LMS algoritmasına göre daha az iterasyon kullanıldığı için daha kısa sürede gerçekleşmektedir. Fakat CM algoritmasında hata değerinin bulunması LMS algoritmasına göre daha fazla işlem yüküne sahip olduğundan bir iterasyon süresine bakıldığında LMS algoritmasına göre daha uzun olduğu fark edilir. Bir iterasyon 20600ps =0.0206 microsaniyeye

karşılık gelir. 17 iterasyon devam ettiğinden dolayı 17x0.0206=0.3502 micro saniye doğru açıyı veren 17 iterasyon çalıştırılma süresidir. Aslında aynı iterasyon devam eden her iki algoritma karşılaştırıldığında LMS algoritması daha kısa sürede tamamlanacaktı. Fakat ortamdan işaretlerin rastgele alındığı farzedildiğinde bu karşılaştırma zamanının değişebileceği kolaylıkla fark edilir.

Son güncellenen ağırlık vektörü sonuçları hem FPGA hem de matlab sonuçları ile karşılaştırıldı. Öncelikle matlabdan elde edilen sonuçları incelenecek.

```
w
ans =

   1.6769 + 0.1199i     1. anten ağırlık reel ve sanal
   2.1158 - 0.0350i     2. anten ağırlık reel ve sanal
   1.6498 - 0.0313i     3. anten ağırlık reel ve sanal
   2.0926 + 0.1093i     4. anten ağırlık reel ve sanal
   1.7175 - 0.1685i     5. anten ağırlık reel ve sanal

w
ans =

1.6769 + 0.1199i     1. anten ağırlık reel ve sanal
 2.1157 - -0.035i     2. anten ağırlık reel ve sanal
 1.6497 - 0.0313i     3. anten ağırlık reel ve sanal
 2.0926 + 0.109 i     4. anten ağırlık reel ve sanal
 1.7175- -0.1685i     5. anten ağırlık reel ve sanal
```

FPGA' de elde edilen son ağırlık değerleri matlabdaki değerlerle karşılaştırıldığında LMS'de olduğu gibi biribirine yakın değerler verir. Burada elde edilen işaretlerin ya birbirinin aynısı olduğu yada binde 1 veya 2 farklı olduğu görülür. Tabi ki ortam koşulları değiştiği taktirde artan iterasyon sayılarında hatanın artması söz konusudur. Çünkü ortamda bulunan girişim değerleri, çokluyollar ve de gürültü sürekli değişebilen yapıya sahiptir. Genel anlamda bakıldığındai CM algoritmasının LMS algoritmasına göre daha iyi açı değerleri bulduğu ortaya çıkar.

Son olarak CM algoritmasının kaynak kullanımı açısından virtex2 ve virtex4 kullanılarak karşılaştırılması Tablo 5.3' de gösterilmiştir.

Tablo 5.3: CM algoritmasının (seri mimari)Virtex2 ve Virtex4 için sentez sonuçları

Kullanılan FPGA	Virtex2(xc2v3000)	Virtex4(xc4vlx60)
Algoritma	**CM**	
Dilim sayısı	14,336'dan 6,534'sı %45	26654'dan 6,313'i %23
Flip-Flop sayısı	28,672'den 4,134'si %14	53248'den 4,134'si %7
4 girişli LUT sayısı	28,672'den 8,225'sı %28	53248'den 7,860'i %16
Giriş çıkış sayısı	484'dan 321'i %66	448'dan 321'i %71
GCLK sayısı	16'dan 1'i %6	32'dan 1'i %3

Not: Ayrıca virtex4 DSP48 bölümünü 64'ün 16'sını %25 oranında kullanır.

Yine LMS algoritmasında olduğu gibi 16 bitlik paralel mimari kullanılarak süre açısından önemli kazanımlar elde edilebilir. Fakat sonuçların elde edilmesinde aynı LMS algoritmasında olduğu gibi küçük sayılar geldiğinde problem çıkabilir. Bu sebeple bu problem çözülmesi halinde süre açısından elde edilebilecek avantajlar incelenecek. Tabi ki LMS algoritmasında olduğu gibi kapasite açısından seri kullanıma göre daha fazla yer kaplayacaktır.

Paralel çalışmada bir iterasyon 2400 pico saniye= 0.0024 micro saniye sürer. Toplam süre karşılaştırıldığında ise 17 iterasyon için 17x0.0024=0.0408 microsaniye olarak bulunur.

Süre açısından, iyi sonuçlar vermesine rağmen kapasite kaynak kullanımı açısından paralel mimari yalnızca 16 bitlik işlemleri gerçekleştirebilir. Bu ise hassasiyeten kaynaklanan hataların oluşmasına neden olur. Eğer kullanıcı hatanın en az olmasını istiyorsa süreden taviz verecek ve de 32 bitlik kullanım ile seri olarak algoritmaları gerçekleyecektir. Bizde özellikler 32 bitlik algoritma üzerinde duruyoruz. Fakat bir sonraki SCC algoritması işlem karmaşıklığı nedeniyle yalnızca 16 bit gerçeklenebileceğinden 16 bitlik süre ve kaynak kullanımı da gösterilmiştir.

Tablo 5.4: CM algoritmasının (paralel mimari) Virtex2 ve Virtex4 için sentez sonuçları

Kullanılan FPGA	Virtex2(xc2v3000)	Virtex4(xc4vlx60)
Algoritma	**CM**	
Dilim sayısı	14,336'dan 8,262'sı %57	26,654'dan 9,701'i %36
Flip-Flop sayısı	28,672'den 2,161'si %7	53,248'den 2,159'si %3
4 girişli LUT sayısı	28,672'den 15,342'sı %53	53,248'den 18,056'i %33
Giriş çıkış sayısı	484'dan 161'i %33	448'dan 161'i %35
GCLK sayısı	16'dan 1'i %6	32'dan 1'i %3

Not: Virtex4'de bulunan DSP48 bölümünün 64'de 42'si %65 oranında kullanılır.

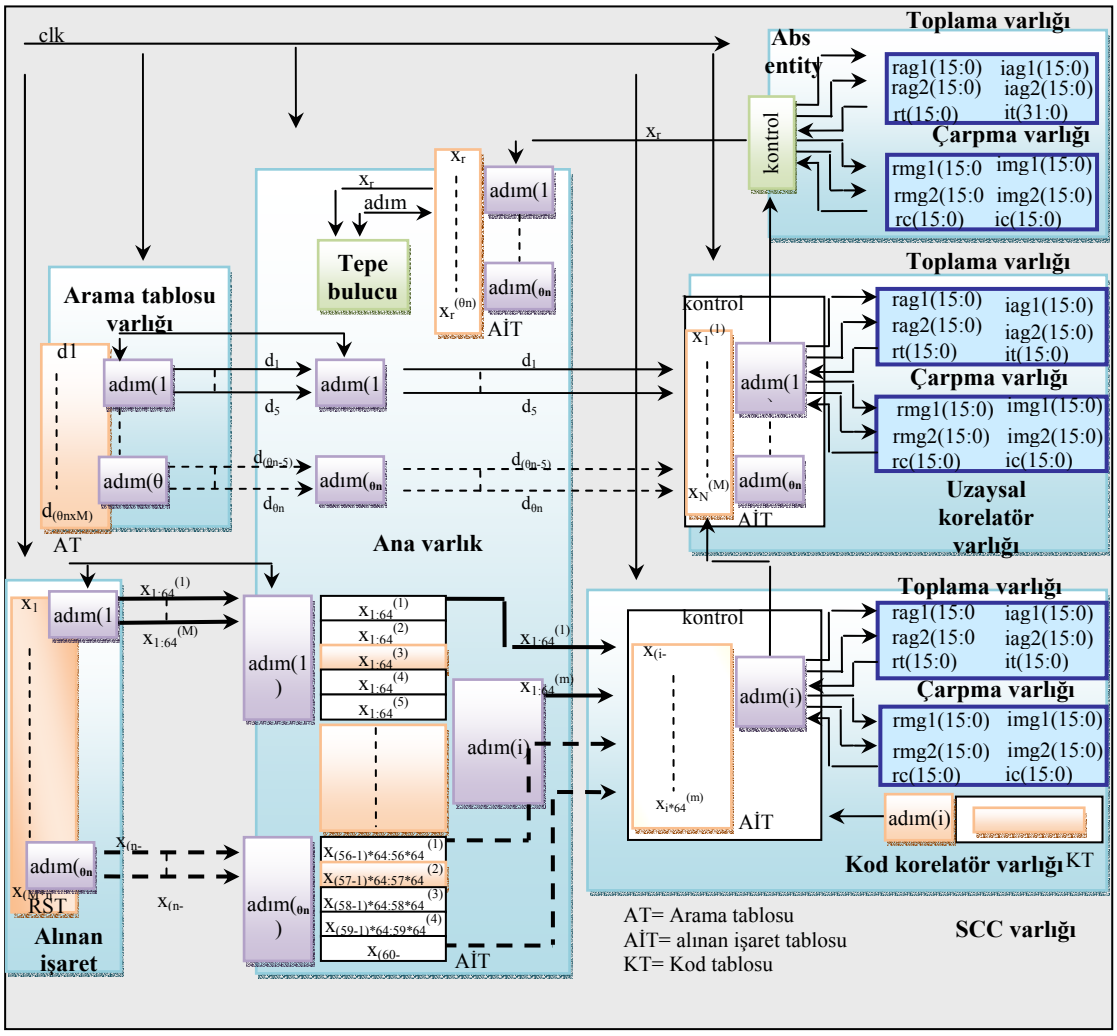
CM algoritmasının Virtex2 ve Virtex4 sentez sonuçları Tablo 5.4' de verilmiştir.

5.3. SCC Algoritmasının FPGA Tabanlı Gerçeklenmesi

SCC algoritmasında da LMS algoritmasında olduğu gibi IEEE 754 formatında işaret değerleri kullanılır. Bu işaret cdma2000 formatında veri oranı 1.288 mcps olacak şekilde üretilir. Gürültü SNR değeri 20dB olup, istenen işaret ve de haricinde 10dB bastırılmış girişim işaretleri kullanılmıştır.(Ayrıca bastırılmış olan girişim işaretlerinden biri çokluyol olarak düşünülmüştür) Burada önemli diğer bir nokta ise 16 bitlik kayan noktalı sayı kullanılmasıdır. LMS ve CM algoritmasının 32 bit olarak gerçeklendiği yukarıdaki bölümde görülmektedir. Bu işlemlerin daha iyi sonuçlar vermesi açısından oldukça önemlidir. Fakat 32 bitlik işlemler SCC, LMS ve de CM algoritmalarına göre çok daha karmaşık yapıya sahip olduğundan ve de işlem yükü çok fazla olduğundan dolayı virtex2 ve virtex4 kullanılarak gerçeklenememiştir. Yalnızca virtex4' de 16 bitlik işaretler ancak %99' lara varan kaynak kullanımı ile gerçeklenebilmiştir. VHDL içinde yazılan kod Xilinx ISE 7.1 kullanılarak herbiri bit dosyasına dönüştürülür. Ve de Virtex4 FPGA'e yüklenir. Burada, Mx1 karmaşık değerlikli ağırlık vektörüdür. M dizi içerisindeki anten elemanlarının sayısını gösterir. Biz FPGA üzerinde M paralel ünitelerine sahip olabiliriz (her toplama ve çarpma üniteleri içererek). FPGA üzerinde büyüklük sınırlandığından her anten eleman değeri içinde paralel ağırlık hesaplaması yerine seri yapılması daha iyi sonuçlar verecektir. Çünkü, seri işlem yapıldığı taktirde ancak FPGA kaynakları yeterli

olacaktır. Özellikle işlem yükünün hafifletilmesi için modül kullanımı birçok avantaja sahip olacaktır. FPGA üzerinde uygulama blokları olarak Ana Varlık, Arama Tablosu Varlığı, Uzaysal Korelatör Varlığı, Kod Korelatör Varlığı, Alınan İşaret Varlığı ve Mutlak Değer Varlığı bulunur.

Şekil 5.7' de SCC algoritması modüler olarak verilmiştir.



Şekil 5.7: SCC algoritması

Bu kısım incelenirse, Kod Korelatörde despread işlemi gerçekleştirmek amacıyla 60 defa 64' lük çarpma işlemi gerçekleştirilir. 64' lük çarpma işleminin tek tek çarpma modülleri kullanılarak gerçekleştirilmesi kaynakların yetersiz kalması anlamına gelecektir. Bu sebeple 64' lük çarpma işleminin gerçeklenebilmesi için 8' lik çarpma

işlemi kullanılır. Onun haricinde uzaysal korelatör işlemi için de aynı tarzda 90 adet açının bulunmasını kolaylaştıracak açı bulma modülü kullanılır. Açı bulma modülünde 5 antene ait değerler sırasıyla alınacağından dolayı burada 5' lik çarpma modülü 5 'e 5' lik matrislerin kolaylıkla çarpılabilmesini sağlar. Bu modüller daha detaylı bir şekilde anlatılacaktır.

Alınan işaret varlığı : SCC varlığının girişleri gibi gelen işaretler için bir arayüz sağlar. Normalde cdma2000 formatına göre işaretimiz 24576 birim değere sahiptir(64x384). 5 anten kullanıldığı göz önüne alındığında 5x24576'lık matris değeri karşımıza çıkar. Bu kadar işlem yapmak sistemin karmaşıklığını oldukça arttıracaktır. Bu yüzden 5x(64x12) değerle işlem yapmamız algoritmanın doğru sonuçlar vermesi için yeterli olacaktır. İşaretlerin gerçek ve sanal kısımları 32 birim büyüklüğünde bir tablo içinde depolanır. M=5 anten için N=12 n=60 adımda başarılır.

Kod korelatör varlığı: Bu varlık istenen kullanıcı ile gelen işaret örneklerinin korelasyonunu gerçekleştirir. Burada alınan işaret varlığını herbir anten için 64x12 (768) değer olarak söyledik. 5 anten için 5x768 (5x64x12)'lik bir alınan işaret matrisi ile 64 birim değere sahip korelasyon vektörü ile ilişkilendirme işlemi yapılır. Bunun için ilk olarak 1. antene ait olan ilk 64 birimlik vektör ile korelasyon vektörü 64 birim çarpılır. Ve de 1. antene ait olan ilk değer elde edilir. Her anten için bu işlem tekrarlanarak 5x1'lik bir vektör elde edilir. Daha sonra yine ilk antenden başlayarak 5. antene kadar aynı işlemler 12 defa tekrarlanarak 5x12'lik bir matris elde edilir. Toplam 60 adımın herbirinde ilişkilendirilmiş (despread) işaret uzaysal korelatör varlığına gönderilir. Burada kod korelatör işleminin yükünün hafifletilebilmesi için sekizlik çarpma işlemi yapılır. Böylece 64 x64' lük matris işleminde 64 defa toplama veya çarpma modülü çağırmaktan ziyade 8x8' lik çarpma işlemi 8 defa çağrılır. Aslında kod korelatör kısmını 64x64 matris çağırma olarak düşünülebilir. Ana programdan sayaç kullanımı ile seri olarak antenden alınan işaret ile istenen işaret çarpılır.

$$\begin{bmatrix} \begin{pmatrix} 1 \ldots .64(1) & \cdots & 1 \ldots \ldots .64(12) \\ \vdots & \ddots & \vdots \\ 1 \ldots \ldots 64(5) & \cdots & 1 \ldots \ldots .64(12) \end{pmatrix} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ . \\ 64 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 & \cdots & 12 \\ \vdots & \ddots & \vdots \\ 5 & \cdots & 60 \end{bmatrix} \end{bmatrix}$$

Arama tablosu varlığı : Yazılan algoritmada, her bir anten değeri için 90 adet açı değeri bulunur. Bu, anten değerlerinin karmaşık sayı değerlerinden olması gerektiği düşünülür. Toplam 5x90 (450) adet karmaşık ağırlık değeri bulunur. Bu varlık, ana varlığa 90 adımda M grupları gibi θ_nxM veri gönderir. (M 5'e karşılık gelir.)

Uzaysal Korelatör Varlığı : Kod korelatör çıkışında, 5x12' lik bir matris üretilmişti. Yine aynı şekilde arama tablosunda 5x90'lık bir matris verilmişti. Buradaki işlemde ise öncelikle, her antene ait ilk değer 1x5'lik matris ile 5x12'lik kod korelatör matrisi çarpılır. Sonuç olarak, 1x12'lik vektör elde edilir. Genel olarak, bu işlemi MxN_s boyutunda dizi matris cevabı ile 1xM veri korelasyonunu başarır şeklinde özetleyebiliriz. Her adım içerisinde, N_sx1 (N_s'e 5 anten sayısına denk olur) boyutunda korelasyon sonucu mutlak değer varlığına gönderilir.

Mutlak değer varlığı : N_sx1 olarak elde edilen uzaysal korelatör çıkışı kendisinin tranpozesi ile çarpılarak matematiksel olarak mutlak değer işlemi elde edilmiş olunur. Bu işlem, her anten için 90 açı değeriyle beraber tekrarlanır. Toplam θ_n (90) adımda her anten için verinin mutlak değerini hesaplar. Bu ifade, ise gücün bulunması anlamına gelmektedir. Genel olarak uzaysal korelatör işlemi mutlak değer varlığını da kapsamaktadır. Burada işlem yükünün azaltılması için yine kod korelatör işleminde olduğu gibi 5x5'lik çarpma işlemini gerçekleştiren 5'lik çarpma kullanılır.

Ana varlık : Bu varlık bir FPGA üzerindeki bütün varlıklar için denetleyici gibi davranır. Bu varlık arama tablosu içindeki uzaysal korelatör varlığından gelen alınan veriyi depolar. Ana varlık içinde bulunan tepe noktası bulucu varlık θ_n veri boyunca en büyük değerin bulunmasını sağlar. Böylece tepe noktası bulucu çıkışında uzaysal korelatörün indeksine karşılık gelen DOA bulunur.

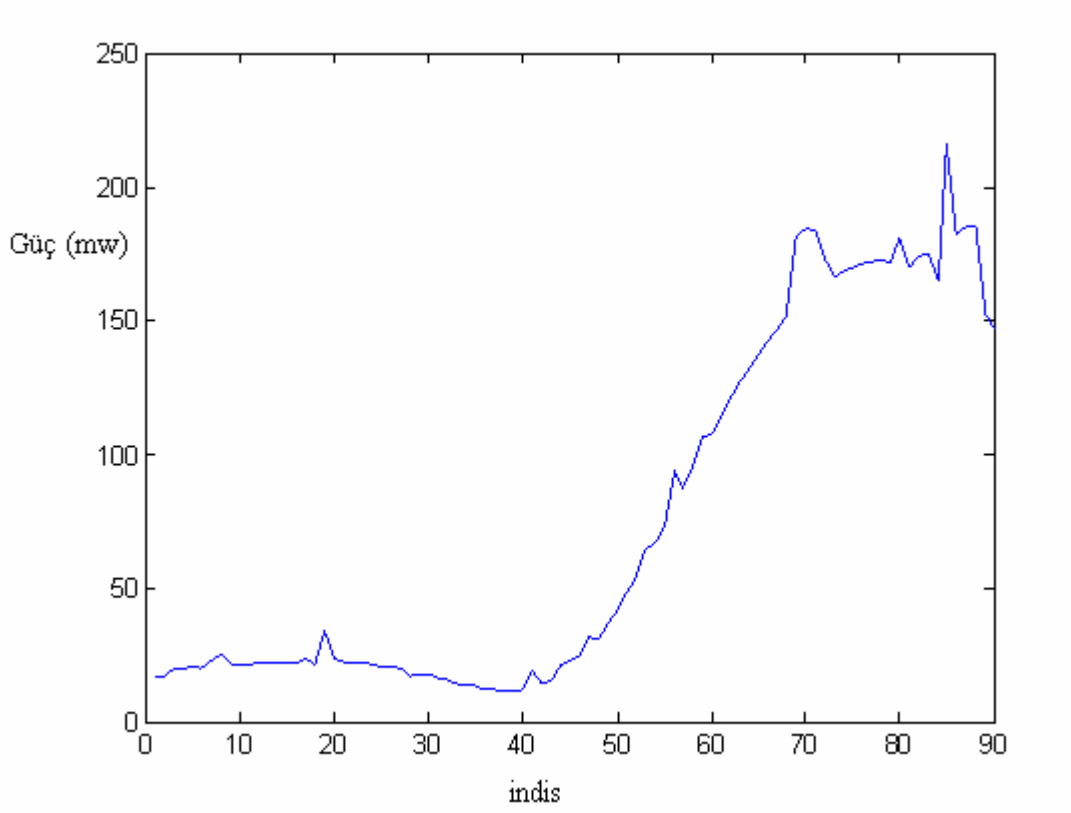
Genel olarak SCC algoritması şu şekilde özetlenebilir.

İlk olarak 5 anten kullanarak ve de cdma2000 formatında 24576 büyüklüğünde veri kullanarak işleme başlanılır. Fakat işlem karmaşıklığı IEEE 754 kayan noktalı sayı formatı kullanıldığı için çok fazla olduğundan dolayı 12 iterasyon olarak işlem

gerçekleştirilir. Bu yüzden 64x12 yani 768’ lik veri kullanarak işlem gerçekleştirilir. Bu doğru sonuçlar vermesi için yeterli olacaktır.

Burada, her anten için de bu işlemi düşünürsek, 5x64x12’ lik bir veri yükü bulunur. Öncelikle, kod korelatör işlemi yapmak için her antenin ilk 64’ lük kısmı istenen kullanıcı ile çarpılır. Böylece, her anten için ilk işlem sonucu skaler bir değer elde edilir. Daha sonra 12 defa 64’ lük matrisler için aynı işlem tekrarlanır ve sonuç olarak da her anten için 12 değer elde edilir.(toplam 5x12 60 değer elde ederiz). Kod korelatör çıkışında 60 değer bulunur.

Bir sonraki adım ise uzaysal korelatör kısmıdır. Bu kısımda, yine kapasiteden uygun bir şekilde yararlamak amacıyla alt modül kullanımı mevcuttur. Bu kısımda kod korelatör çıkışında elde edilen 5x12’ lik değer ile arama tablosunda bulunan 5 antene ait olan 90 açı değeri ile işlem yapılır. İlk arama tablosunda, her antene ait ilk açı değeri ile kod korelatör çıkışında elde edilen 5x12’ lik matrisin sırayla ilk sütundan itibaren 12. sütuna kadar çarpılır. Burada 1x12’ lik matris elde edilir.1x12’ lik matris kendisinin transpozesi ile çarpılarak skaler bir değer elde edilir. Bu skaler değer ilk açı değerindeki güce karşılık gelmiş olur. Maksimum güç değeri ise açı kestirimi için yardımcı olur. Bu işlem, her antene ait 90 açı değerinin kod korelatör ile çarpımı ile tekrarlanır. Bu sebeple her bir açının bulunması için açı bulma modülü kullanılması faydalı olur. Yukarıdaki işlemler incelendiğinde kod korelatör çıkışının 5x12’ lik bir matris ve de her antene ait ayrı ayrı 90 açı içereren arama tablosunun 5x90’ lık matrise karşılık gelmesinden dolayı 5x5’ lik matris çarpma işlemi yapabilen beşlik çarpma modülü kullanılabilir. Bu kod korelatöre yardım amacıyla kullanılan sekizlik çarpma modülü gibi işlem yükünü oldukça hafifletir. Son kısımda ise 90 açı değerine karşılık gelen güç değerlerinin karşılaştırılmasıdır. Bu kısımda sırayla güç değerleri karşılaştırılır. En yüksek güç değerine sahip olan indis bulunur. İşlem yükünün çok fazla olmasından 2 derece açı aralığı ile düzgün doğrusal dizi 90 açı değeri ile gösterildiğinden bulunan indisin 2 katı alınır.



Şekil 5.8: SCC algoritması için indis değerlerine karşılık gelen güç değerleri

Şekil 5.8' den de anlaşılacağı gibi 85. indis değerine karşılık gelen 85x2=170 derece açı kolaylıkla fark edilebilir. Burada, girişim işaretinde aynı şekilde yüksek güç değerlerine sahip olduğu kolaylıkla görülebilir. Özellikle, teorikte istenilen işaretin matlabda algoritma ile elde ettiğimiz ve de FPGA kullanarak bulduğumuz sonuçlarla karşılaştırdığımızda istenen işaret 169 derece olarak rastgele üretilmiştir. Aynı şekilde, 10dB zayıflatılmış bir girişim işareti ve 10dB zayıflatılmış bir çokluyol işareti kullanılmıştır. Ortamda SNR=20dB olan gürültü bulunmaktadır. Algoritma matlabda 169 derecelik açıyı 170 derece olarak yakalayabilmektedir. FPGA'deki sonuçları incelediğimizde ise 168 derecede sonucu yakalayabilmektedir. Buradan da anlaşıldığı gibi matlab ile FPGA arasında bir indis fark etmektedir. Güç değerleri bu iki indis arasında birbirine çok yakın olduğundan istenen işarette bu iki indis arasında olduğundan bu sonuçlar ortaya çıkmaktadır. Şekil 5.8' den de anlaşıldığı gibi farklı noktalarda oluşan girişim ve gürültü işaretin farklı açı değerlerinde yüksek güç değerlerine sahip olmasını sağlar. Burada 70. indis değerine karşılık gelen 140 derecede ve 88. indis değerine karşılık gelen 176 derecede girişim işaretleri görülür.

SCC algoritması oldukça büyük işlem yüküne sahiptir. IEEE 754 kayan noktalı sayı formatının kullanılması ve de çok fazla matris işleminin bulunması SCC algoritmasının FPGA' de gerçeklenmesini zor kılmıştır.

SCC algoritmasının süre olarak performansını incelendiğinde işlem yükü LMS ve CM algoritmalarına göre çok daha fazla olduğundan SCC algoritması yapılan çalışmada daha uzun zaman harcamaktadır. Burada adım adım gidilerek süre hakkında bilgi verilecek.

İlk olarak kod korelatörü incelersek, 70100 pico saniye=0.0701 micro saniye bir anten ait olan ilk 64 değer ile 64 değere sahip olan istenen işaretin çarpılması ile elde edilir. Bu işlem kod korelatör işlemi için 5 antene ait 12 tane 64' lük değer için 60 adımda gerçekleştirilir. Sonuç olarak kod korelatör işlemi 60x0.0701= 4.201 micro saniye sürer.

Uzaysal korelatör kısmında ise kod korelatör çıkışında elde edilen 5x12' lik matris ile arama tablosundaki değerler işleme sokulur. Burada, 90 açı değerinden her birinin güç değeri 76700 pico saniye= 0.0767 micro saniye de elde edilir. Bu işlem ise 90x0.0767= 7.803 micro saniye olarak hesaplanır. FPGA' de dikkat edilmesi gereken nokta bu modüllerin senkronize çalışması gerekliliğidir. Çünkü, modüller clock darbesi ile çalıştıkları için modülde yapılan işlem bittikten sonra sayac sıfırlanmalı ve de bir sonraki adım tekrar sıfırlanmış sayaç ile başlamalıdır. Birbirinde bağımsız modüller ve bu modülerlerde onların işlem yükünü azaltacak alt modüller kullanıldığı için senkronizasyon SCC algoritmasında önemli bir değere sahiptir. Bu sebeple, programın doğru çalışabilmesi için toplam sürede taviz verilmiştir. Aynı zamanda, son kısımda en büyük güç değerinin bulunabilmesi için sayaç kullanılarak seri karşılaştırma işlemi yapılmıştır. Algoritma toplam 11247900 pico saniye=11.2479 micro saniyede tamamlanır.

LMS ve CM algoritmaları iteratif yapıya sahip olduğundan değişen ortam koşullarında algoritmanın iterasyonu oldukça uzun sürebilir. Bu da LMS ve CM algoritmasının sürelerinin artması anlamına gelir. Aynı zamanda CM algoritmasının

yakınsama problemi ve de her iki algoritmanın da çok kötü kanal şartlarında etkisiz kalabilmesi SCC algoritmasını diğer algoritmalara göre daha önemli kılar. SCC algoritmasının daha hassas sonuçlar verebilmesi için çeşitli alternatifler kullanılabilir. Uzaysal korelatörde 2 derecelik ara ile 180 derecelik düzgün doğrusal diziye sahip açı tanımlanmıştı. Burada kodun optimize edilmesi ile veya daha iyi performansa sahip FPGA' lerin kullanılması ile 180 veya 360 kısıma ayrılabilecek açı aralığı ile daha iyi sonuçlar elde edilerek açı değeri yakalanabilir. Bunun haricinde, bir diğer önemli faktör ise 16 bitlik kayan noktalı sayı kullanımı yerine 32 bitlik kayan noktalı sayı kullanımıdır. Fakat, kullanılan FPGA' de 16 bitlik kayan noktalı sayı kullanımına rağmen kapasitenin %99 civarında olması yine ancak kodun optimize edilmesi durumunda veya daha iyi bir FPGA kullanılması halinde 32 bitlik kayan noktalı sayı kulanılabileceğini gösterir.

Tüm bu faktörler göz önüne alındığında SCC algoritması daha uzun sürede gerçeklenmesi aynı zamanda, çok büyük kaynak kullanımı dezavantajlarına rağmen ortam şartlarından LMS ve CM algoritmalarına göre daha az etkilendiği için önerilmiştir. Aynı zamanda, daha hassas sonuçlar alınabilmesi de değişiklikler yapılabilmesi söz konusudur. Bir de SCC algoritması kaynak kullanımı incelenecek. Fark edildiği üzere dilim sayılarının tamamına yakını kullanılır. Yapılan çalışma ancak virtex4 FPGA ile gerçeklenebilmiştir. FPGA simulasyon olarak gerçekleştirilmesine rağmen virtex2 FPGA' e sentezlenemez. Öncelikle bu algoritmaların kaynak kullanımı bakımından karşılaştırılması Tablo 5.5.' de gösterilmiştir.

Tablo 5.5. SCC algoritmasının kaynak kullanımı

	SCC
Kullanılan FPGA	Virtex4 xc4vlx60
Dilim sayısı	26,624'dan 26,622'sı %99
Flip-Flop sayısı	53,248'den 10,727 %20
LUT sayısı	27392'den 43,560'sı %81
Giriş çıkış sayısı	17'dan 448'i %3
GCLK sayısı	32'dan 1'i %3
DSP48 sayısı	64'dan 8'i %12

Bu üç algoritmanın kaynak kullanımı açısından değerlendirilmesi Tablo 5.6' dan görülebilir. Tablodan da görüldüğü gibi LMS ve CM algoritmasının kaynak kullanımı miktarı biribirine çok yakın iken SCC algoritması yaklaşık iki katı büyüklüğünde kaynak kullanımına sahiptir [44].

Tablo 5.6: Adaptif algoritmaların sentez sonuçlarının karşılaştırılması

Virtex4 xc4vlx60			
Adaptif Algoritmalar	**LMS**	**CM**	**SCC**
Dilim sayısı	26,654'dan 6762'sı %25	26654'dan 6,313'i %23	26,624'dan 26,622'sı %99
Flip-Flop sayısı	27392'den 3921'si %7	53248'den 4,134'si %7	53,248'den 10,727 %20
LUT sayısı	53,248'den 9,203'sı %17	53248'den 7,860'i %16	27392'den 43,560'sı %81
Giriş çıkış sayısı	448'dan 321'i %77	448'dan 321'i %71	17'dan 448'i %3
GCLK sayısı	32'dan 16'sı %11	32'dan 1'i %3	32'dan 1'i %3

Tablo 5.7' da ise daha önce her algoritma anlatılırken belirtilen süreler tablolar halinde görülmektedir. Aynı zamanda, daha önceden DSP kullanılarak gerçekleştirilen bu algoritmaların zaman performansı Tablo 5.8' de verilmiştir.

Tablo 5.7: Algoritmaların sentez sonuçlarının gerçeklenme sürelerinin karşılaştırılması

Algoritmalar			
FPGA Kullanımı	**LMS**	**CM**	**SCC**
Seri	0.6072 μ s (32 bit)	0.3502 μ s (32 bit)	11.2479 μ s (16 bit)
Paralel	0.059 μ s (16 bit)	0.0408 μ s (16 bit)	_____

Tablo 5.8. Algoritmaların DSP ve FPGA sentez sürelerin karşılaştırılması

			LMS	CM	SCC
ortalama	DSP	C6701 C6711 C6713	10.46ms 9.97ms 6.63ms	11.02ms 10.31ms 7.39ms	10.75ms 12.15ms 9.28ms
	FPGA	Seri Paralel	0.6072 μs 0.059 μs	0.3502 μs 0.0408 μs	11.2479 μs ---------- -

Daha önce, aynı algoritmalar için yapılan DSP çalışması ile yaptığımız FPGA çalışmasının sonuçlarını da karşılaştırdığımızda FPGA ile elde edilem çalışmaların süresinin DSP ye göre çok daha küçük olduğu rahatlıkla bulunabilir [45].

Burada, önemli bir nokta kaynak kullanımı ve hız faktörlerinden hangisinin bizim için önemli olduğudur. FPGA'lerin en önemli tercih sebeplerinden biri hızlarının diğer işlemcilere göre çok iyi olmasıdır. Bu ise kaynakların paralel kullanımı ile mevcuttur. Eğer tasarladığımız sistemde hız önemli olsun istiyorsak ki akıllı anten algoritmaları için gezgin istasyon sürekli hareket halinde olabileceğinden en önemli faktör kaynakların paralel olarak kullanılmasıdır. Seri işlemlere göre çok daha fazla kaynak kullanımı olabileceğinden kapasite artacaktır. Fakat kapasite artışı bizim için bir dezavantaj getirmiyor ise, kapasite sınır noktaya kadar aşmamak şartıyla kullanılabilir.

BÖLÜM 6. SONUÇ ve İLERİ ÇALIŞMALAR

Yapılan çalışma, gelecek nesil kablosuz haberleşme sistemlerinden olan LMS, CM ve SCC gibi akıllı anten algoritmalarının FPGA üzerinde gerçeklenmesini sağlar. Cdma2000 standartına uygun olan üretilen işaret Xilinx ailesine ait olan Virtex II ve Virtex 4 FPGA' ine uygulanmıştır. Kullanılan işaret modeli için Düzensiz Doğrusal Dizi (Uniform Lineer Array, ULA) düşünülmüş olup, değişken çokluyol ve girişim özellikleri varsayılır.

Bu çalışmada, bu algoritmaların hesaplama zamanları, kaynak kullanımı ve sonuçlara göre hata karşılaştırılması verilmiştir. Bir önceki bölümdeki kaynak kullanımı sonuçlarından da görüldüğü gibi LMS ve CM algoritmalarına göre çok daha fazla matematiksel işlem karmaşasına sahip olan SCC algoritmasının kaynak kullanım oranı bu iki algoritmaya göre çok daha fazladır. Aynı zamanda SCC algoritması çok karmaşık bir yapıya sahip olduğundan yalnızca 16 bit kayan noktalı sayı kullanımıyla algoritma gerçekleştirilir. Kaynak kullanımı dikkatli bir şekilde incelenirse, SCC algoritmasının Virtex4 FPGA'i %99' lara ulaşan kapasite ile kullandığı kolaylıkla farkedilir. Virtex 2 FPGA kapasitesi SCC algoritmasının gerçeklenmesi için yetersizdir. LMS ve CM algoritması ise hem Virtex2 hem de Virtex4 FPGA'e sığdırılabilir. Burada paralel kullanım ile sayaç kullanılarak seri uygulama mevcuttur. Paralel mimari FPGA' in temel çalışma prensibi olduğundan dolayı hız açısından seri mimariye göre yaklaşık 10 kat iyi sonuçlar elde edilir. Fakat buradaki problem seri mimari kullanımı sayaçlar kullanıldığı kaynakların tasarruflu kullanımı anlamına gelir. Bu sebeple 32 bit kayan noktalı sayı formatı rahatlıkla kullanılabilir. Bu kullanım %0.2 veya tam (hatasız) sonuçların elde edilmesi anlamına gelir. Fakat, her ne kadar hız açısından paralel mimari kullanılsa bile, yetersiz kapasite sebebiyle yalnızca 16 bit kayan noktalı sayı kullanımı mevcuttur. 16 bit kayan noktalı sayı kullnımı da hatalı sonuçlar verebilir. Bu sebeple, yapılan çalışmada 32 bitlik seri mimari (sayaç kullanımı) tercih edilmiştir.

Bu algoritmaların gerçeklenme süreleri karşılaştırıldığında ise örnek olarak 100MHz clk olduğu düşünülürse, LMS ve CM algoritmalarında birbirine yakın sonuçlar içerirken SCC algoritması çok daha uzun sürede gerçekleşir. Fakat, SCC algoritması istenen işaretin yakalanabilmesi için daha iyi sonuçlar vermektedir. LMS algoritması hata hesaplanmasında pilot işaret kullandığından hatanın hesaplanmasında çıkış dizisinin hesaplanması (daha fazla işlem yükü) mantığına dayanan CM algoritmasına göre bir iterasyon düşünüldüğünde daha hızlıdır. Fakat, rastgele üretilen alınan işaret değerlerine göre CM algortiması açıyı daha az iterasyonda bulduğundan genel süre olarak daha iyi sonuçlar elde edilir. Ama anten den alınan işaret değerleri sürekli değişebileceğinden süre de buna bağlı olarak değişebilir. SCC algoritması ise işlem yükü çok daha fazla olduğundan yaklaşık 20 kat daha uzun sürede gerçekleşir.

Aynı zamanda, daha önce DSP üzerinde yapılan çalışmada elde edilen gerçeklenme süreleri ile karşılaştırıldığında FPGA'lerde paralel çalışma mimarisine sahip oldukları için gözle görülür farklar elde edilmektedir. DSP ile ms' ler mertebesinde süre sonuçları elde edilirken, FPGA' de bu sürenin μs' ler mertebesinde olduğu rahatlıkla görülebilir.

Kapasite açısından paralel mimari kullanımı ile gerçekleştirilen LMS ve CM algoritmalarının seri mimariye göre gerçekleştirilen LMS ve CM algoritmalarına göre iki kat fazla kaynak kullanımına sahip olduğu görülür. Burada basit bir örnek ile eğer paralel mimari kullanırsak algoritmaların yalnızca birini FPGA' e yükleyebileceğimiz fakat seri mimari kullanılarak (süre daha kötü) her iki algoritmanın yalnızca bir FPGA'e sığdırılabileceği sonucuna varılabilir. SCC algoritmasında ise yalnızca seri mimari kullanımı mevcut olup, kapasite tamamen kullanılır.

Bu çalışmadan da anlaşılacağı gibi FPGA' ler hız performasından gelecek nesil kablosuz haberleşme sistemlerinin vazgeçilmez parçasıdır. DSP daha iyi kapasiteye sahip olmasına rağmen FPGA süre açısından çok daha iyi sonuçlar verir. Karmaşık uygulamalarda kapasite açısından, DSP ve hız açısında FPGA kullanımı tercih edilmesine fayda vardır. Günümüzde kullanılan bazı FPGA' ler (Virtex4) hem DSP hem de FPGA içermektedir.

Aynı zamanda, günümüzde akıllı antenlere paralel olarak çoklu giriş çoklu çıkış (Multiple Input Multiple Output, MIMO) ve Wimax Baz İstasyonu tasarımı gibi çalışmalarda geliştirme faaliyetleri devam etmektedir.

İleride yapılacak çalışmada daha farklı gelecek nesil sistemler için farklı algoritmalar kullanılarak performans karşılaştırılması düşünülmektedir.

KİŞİSEL YAYINLAR ve ESERLER

A. Uluslararası bilimsel toplantılarda sunulan ve bildiri kitabında basılan bildiriler:

1. Sener Dikmese, Adnan Kavak, Suhap Sahin, Kerem Kucuk, and Hasan Dincer, ; Evaluation of FPGA-based Software Radio Beamformers for 3G Wireless;, IEEE Radio and Wireless Sysmposium 2007 (RWS2007), will be published, January 9-11, 2007 Long Beach, CA, USA. (IEEE Xplore)

2. Suhap Sahin, **Sener Dikmese**, Kerem Kucuk and Adnan Kavak ;A Comparative Study of Antenna Array Algorithm Implementations using FPGA and DSP for cdma2000 ;, 3rd International Symposium on Wireless Communication Systems 2006 (ISWCS2006), will be published, September 5-8, 2005 Valencia, Spain. (IEEE Xplore)

B. Ulusal bilimsel toplantılarda sunulan ve bildiri kitabında basılan bildiriler:

3.Sener DIKMESE, Merve DEMIR, Ugur KAFADAR, Hasan DINCER, "Haberleşme Modülasyonlarının FPGA Tabanlı Gerçeklenmesi" XV. SIU-Türkiye 2007 Bilimsel Kongresi, 2007 Anadolu Üniversitesi, Eskişehir.

4. Okan Ilker, **Sener Dikmese** ve Hasan Dıncer, "DSSS'in FPGA'de gerçeklenmesi" XV. SIU-Türkiye 2007 Bilimsel Kongresi, 2007 Anadolu Üniversitesi, Eskişehir.

5. M. Ali Çavuşlu, **Şener Dikmeşe**,Suhap Şahin, Kerem Küçük, ve Adnan Kavak, "Akıllı Anten Algoritmalarının IEEE 754 Kayan Sayı Formatı ile FPGA Tabanlı Gerçeklenmesi ve Performans Analizi";, III. URSI-Türkiye 2006 Bilimsel Kongresi, 2006 Hacettepe Üniversitesi, Ankara.

KAYNAKLAR

[1] F. Gross, "Smart Antennas for Wireless Communications with Matlab",100-250, **McGraw-Hill Companies**, (2005)

[2] P.R.P Hoole., "Smart Antennas and Signal Processing for Communication, Biomedical and Radar Systems" 50-150, **WIT Press,** (2001).

[3] Xilinx Attributes,. "Constraints and Carry Logic"- **printed USA** (1997)

[4] Widrow B., et al., "Adaptive Antenna Systems", **IEEE Proc.,** 55, 22143-2158, (1997).

[5] Litva, J., and T. Kowk-Yeung Lo.,"Digital Beamforming in Wireless Communications", **Artech House,** (1996)

[6] Van Atta, L.,. "Electromagnetic Reflection", **US Patent 2908002,** (1959)

[7] York, R., and T. Itoh. "Injection and Phase- Locking Techniques for Beam Control", **IEEE Transcations on MTT,** vol.46, No.11, pp 1920-1920, (1998) .

[8] Howels P., "Explorations in Fixed and Adaptive Resolution at GE and SURC" IEEE Transactions on Antenna and propagation, **Special Issue on Adaptive Antennas**" Vol.AP-24, No. 5, 575-584, (1976).

[9] Widrow, B ., P. Mantey, L. Griffiths,. "Adaptive Antenna Systems", **Processings of the IEEE,** Vol. 55, (1967).

[10] Reed, I., J. Mallet and L. Brennen ,. "Rapid Convergence Rate in Adaptive Arrays" **IEEE Transaction on Aerospace on Electronics Systems,** Vol. AES-10, 853-863, (1974)

[11] Godara, L., Application of Antenna Arrays to Mobile Communications, Part II: "Beamforming and Direction of Arrival Considerations", **Proceedings of the IEEE,** vol.85, No.8, 1195-1245, (1997).

[12] MITOLLA, J.," Software Radio Architecture", **IEEE Communications Magazine,** vol.33(5), 26-38, (1995).

[13] NAGUIB, A. F., "Adaptive antennas for CDMA wireless networks", Ph.D dissertation, **Stanford University, USA,** (1996).

[14] LIBERTI, J. C. Jr. and RAPPAPORT, T. S., "Smart Antennas for Wireless Communications: IS-95 and Third Generation Applications", **Prentice Hall Upper Saddle River, NJ,** (1999).

[15] Pattan, B., "Robust Modulation methods and Smart Antennas in Wireless Communications", **Prentice Hall, Newyork,** (2000)

[16] Godara, LC., "Error analysis of the optimal antenna array processors", **IEEE Trans. Aerospace Elect. Sys.** 22, 395-409, (1986).

[17] W.L. Stutzman and G.A. Thiele,. "Antenna Theory and Design", **John Wiley & Sons NY**, (1981)

[18] Yıldız, M.G.,"Adaptif Antenlerde Sayısal İşaret İşleyici Uygulamaları", Yüksek Lisans Tezi, **K.T.Ü, Fen Bilimleri Enstitüsü**, Trabzon (2004)

[19] WINTERS, J.H.,. "Smart antennas for wireless systems", **IEEE Pers. Commun. Mag., vol. 5, no. 1**, 23-27. (1998)

[20] AL-JAZZAR, S. and RADHAKRISHNAN, R.,"Smart Antennas in Wireless Communications", Phd thesis, **University of Cincinnati.** (2000)

[21] Chan , G.K., "Effects of sencronization on the spectrum efficiency of cellular radio systems", **IEEE trans. on Vehicular Technology, Vol. 41 no:3** (1997)

[22] Saunders, S.R., "Antennas and Propogation for Wireless Communication Systems", **John Wiley & Sons,** (1999)

[23] WILLERT, C., "The Evolution of Programable Logic Design Technology", **Xilinx Inc.** (2000).

[24] Bob Zeidman,. "An Introduction to FPGA Design" **Embedded System Conference** (1999)

[25] U. Meyer-Base, "Digital Signal Processing with Field Programmable Gate Arrays" **Springer** (2001)

[26] IEEE Std 1076,. "Edition IEEE Standard VHDL Language" **Reference Manual**" (2000)

[27] Mentor Graphics. "VHDL" **Reference Manual** (1999)

[28] Enoch O. Hwang "Microprocessor Design Principles and Practices With VHDL" (2004).

[29] Doulos. "The VHDL Golden" **ReferenceGuide**, (1995)

[30] TIA/EIA"Interim Standard Physical Layer Standard for cdma2000 Spread Spectrum Systems" **TIA/EIA /S-2000-2**, (2000).

[31] Synario Design Automation. VHDL **Reference Manual** (1997)

[32] Xilinx University Program Virtex-II Development System Hardware **Reference Manual** (2005)

[33] Godara,L.C,. “Smart Antennas” **CRC Press** (2004)

[34] Kavak, A.,”Üçüncü Nesil (3G) Gezgin Haberleşme Sistemleri İçin Smart Anten Sistemi Algoritması Geliştirilmesi ve Prototip Üzerinde Uygulanması”, **TUBİTAK Raporu, 102E015,** 30-60 (2005)

[35] Perini, P.L., and C.L.Holloway, Angle and Space Diversity Comprasions in Different Mobile Radio Environments **IEEE Trans. on Antennas and Propagation, Vol. 46, No. 6**, 764-775 (1998).

[36] Kucuk.,” Cdma2000 Yukarı Bağlantı Kanalı için Akıllı Anten Algoritmalarını Dijital işaret işlemciler üzerinde yazılım radio ile gerçekleştirilmesi”, Yüksek lisans tezi, **Kocaeli Üniversitesi Fen Bilimleri Enstitüsü** İzmit, (2005)

[37] Godara,L.C., and D.B.Ward A., “General Framework for Blind Beamforming” **Proc.of the IEEE Region 10 Conference, Vol.2, No**., 1240-1243 (1999).

[38] Godara, L.C and Cantoni, A., “Analysis of constrained LMS Algorithm with application to adaptive beamforming using perturbation sequences”, **IEEE trans. Antennas Propagat.,** 34, 368-379 (1986).

[39] Godara L.C., “Improved LMS Algorithm for Adaptive Beamforming” **IEEE Trans. Antennas Propagat** Vol. 38, 1631-1635 (1990).

[40] Fabre, P and Gueguen, C., “Improvement of the fast recursive least squares algorithms via normalization : a comparative study”, **IEEE Trans. Acoust. Speech Signal Process.,**34-296 308 (1986.).

[41] Chiba, I., Chuo, W. and Fujise , M., “Beamspace constant modulus algorithm adaptive array antennas”, **IEEE 8th Int. Conf. Antennas Propagat**., 975-978, (1993).

[42] IEEE “Standard for Binary Floating Point Arithmetic”, **ANSI/IEEE Std. 754-1985, IEEE.** (1985)

[43] Xilinx, Virtex II 1.5V field-programmable gate arrays **Xilinx Data Sheet**, 1-4. (2001)

[44] Sahin, S., Dikmese, S., Kucuk, K. and Kavak, A.,. “A comparative study of antenna algorithm implementation using FPGA and DSP for cdma2000”, **ICWS,** Spain(2006)

[45] Dikmese, S., Kavak, A., Sahin, S., Kucuk, K. and Dincer H.,. “Evaluation of FPGA-based Software Radio Beamformers for 3G”, **RWS**, USA (2007)

ÖZGEÇMİŞ

11 Ocak 1983 yılında Edirne'de doğdu. İlk ve orta öğrenimini Edirne'de tamamladı. Lise öğrenimine 1997 yılında kazandığı Kocaeli Körfez Fen Lisesi'nde başlayan Şener Dikmeşe 2000 yılında Edirne Süleyman Demirel Fen Lisesi' nden mezun oldu. 2000 yılında kazandığı Kocaeli Üniversitesi, Mühendislik Fakültesi, Elektronik ve Haberleşme Mühendisliği Bölümünden 2005 yılında Elektronik ve Haberleşme Mühendisi olarak mezun olmuştur. 2005 yılı aralık ayında Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümünde Araştırma Görevlisi olarak göreve başlamıştır. 2005 yılında Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Ana Bilim Dalında yüksek lisans programına başlayan Şener Dikmeşe 2007 yılı Haziran ayı itibariyle bu programdan mezun olma durumundadır. Ayrıca Şener Dikmeşe tez çalışmasının son 6 ayını Amerika Birleşik Devletleri South Florida Üniversitesi Kablosuz Haberleşme ve İşaret İşleme Laboratuvarı' nda (Wireless Communication and Signal Processing Lab.) tamamlamıştır.