KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ ALGILAYICI AĞLARDA KULLANILAN MAC PROTOKOLLERİNİN KARŞILAŞTIRMALI BAŞARIM ANALİZİ

YÜKSEK LİSANS

Serdar SOLAK

Anabilim Dalı: Bilgisayar Mühendisliği

Danışman: Yrd. Doç. Dr. Celal ÇEKEN

KOCAELİ, 2008

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ ALGILAYICI AĞLARDA KULLANILAN MAC PROTOKOLLERİNİN KARŞILAŞTIRMALI BAŞARIM ANALİZİ

YÜKSEK LİSANS TEZİ

Serdar SOLAK

Tezin Enstitüye Verildiği Tarih: 07 Ocak 2008

Tezin Savunulduğu Tarih : 06 Şubat 2008

Tez Danışmanı	Üye	Üye
Yrd. Doç. Dr. Celal ÇEKEN	Doç. Dr. İsmail ERTÜRK	Yrd. Doç. Dr. Ahmet Turan ÖZCERİT
(.....)	(.....)	(.....)

KOCAELİ, 2008

ÖNSÖZ VE TEŞEKKÜR

Kablosuz algılayıcı ağların tasarımını etkileyen en önemli sınırlamalardan biri de enerji tüketimidir. Algılayıcı düğüm enerji tüketiminin azaltılmasına yönelik olarak literatürde çok sayıda çalışma bulunmaktadır. Bu çalışmalar, yönlendirme ve MAC protokolleri üzerine yoğunlaşmış durumdadır.

Kablosuz algılayıcı ağlarda kullanılan MAC protokolleri TDMA ve çekişme tabanlı olmak üzere ikiye ayrılır. Bu tez çalışmasında; TDMA yöntemini kullanan EDSMAC protokolü ile çekişme tabanlı standart DCF MAC protokolü tasarım modelleri ve karşılaştırmalı başarım analizi OPNET Modeler yazılımı kullanılarak gerçekleştirilmiştir. Ayrıca standart DCF MAC protokolüne enerji tüketimini azaltmak üzere uyku modu işlevide eklenmiştir.

Çalışmalarım boyunca benden yardımlarını ve tavsiyelerini esirgemeyen değerli danışmanım Yrd. Doç. Dr. Celal ÇEKEN'e teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SEMBOLLER

Kısaltmalar

ACK	: ACKnowledgement
AD	: Algılayıcı Düğüm
BMAC	: Berkeley MAC
CS	: Carrier Sense (Taşıyıcı sezme)
CSMA/CA	: Carrier Sense Multiple Access with Collision Avoidance
CTS	: Clear To Send
CW	: Contention Window
ÇD	: Çıkış Düğümü
DCF	: Distributed Coordination Function
DIFS	: Distributed Coordination Function IFS
E	: Enerji
EED	: End To End Delay (Uçtan uca gecikme)
EDSMAC	: Energy aware and Delay Sensitive MAC
GHz	: Giga Hertz (frekans birimi)
GMAC	: Gateway MAC
IEEE	: Institute of Electrical and Electronics Engineers
IFS	: Inter Frame Space (Çerçeveler Arası Boşluk)
KAA	: Kablosuz Algılayısı Ağlar
MAC	: Media Access Control (Ortama Erişim Kontrolü)
MD	: Merkezi Düğüm
OPNET	: OPtimized Network Engineering Tool
P	: Güç
PDF	: Probability Density Function
RTS	: Request To Send
SIFS	: Shortest IFS
SMAC	: Sensor MAC
SN	: Sensor Node
TDMA	: Time Division Multiple Access
TMAC	: Timeout MAC
WSN	: Wireless Sensor Networks

KABLOSUZ ALGILAYICI AĞLARDA KULLANILAN MAC PROTOKOLLERİNİN KARŞILAŞTIRMALI BAŞARIM ANALİZİ

Serdar SOLAK

Anahtar Kelimeler: Kablosuz Algılayıcı Ağlar, MAC Protokolleri, OPNET, TDMA, CSMA, DCF, Enerji tüketimi.

Özet: Son zamanlarda iletişim teknolojilerinde yaşanan hızlı gelişmelerle beraber, kablosuz algılayıcı ağların kullanım alanları da hızla artmaktadır. Askeri, sağlık, çevresel, ticari v.b. gibi birçok alanda yaygın olarak kullanılmaya başlanmıştır.

Kablosuz algılayıcı ağların tasarımında en önemli sınırlamalardan biri de enerji tüketimidir. Algılayıcı düğümlerin enerji tüketimi, kablosuz algılayıcı ağın yaşam zamanını etkilemektedir. Bu nedenle algılayıcı düğümlerin enerjilerini verimli kullanmaları kablosuz algılayıcı ağlar için büyük önem taşımaktadır. Enerji tüketimini verimli hale getirmek için akademik ve endüstriyel alanda çok sayıda çalışma yapılmaktadır. Bu çalışmalar genellikle yönlendirme ve MAC protokolleri üzerine yoğunlaşmış durumdadır.

Kablosuz algılayıcı ağlar için önerilen MAC protokolleri merkezi ve çekişme tabanlı olmak üzere ikiye ayrılmaktadır. Merkezi tabanlı olarak önerilen MAC protokollerinde TDMA yönteminin özellikleri kullanılırken, çekişme tabanlı olarak önerilen MAC protokolleri çoğunlukla CSMA/CA yöntemini temel almaktadır.

Tez çalışmasında, kablosuz algılayıcı ağlarda önerilen MAC protokollerinin alt yapısını oluşturan TDMA ve CSMA/CA protokolleri detaylı bir şekilde anlatılmıştır. Ayrıca OPNET Modeler yazılımı yardımıyla TDMA yöntemini kullanan EDSMAC protokolü ile çekişme tabanlı enerji duyarlı DCF MAC protokolü modellenerek karşılaştırmalı başarım değerlendirmeleri sunulmuştur.

EDSMAC protokolünün, enerji duyarlı uygulamalarda, enerji duyarlı DCF protokolüne göre yaklaşık olarak 20 ila 85 kat daha az oralama uçtan-uca gecikme verdiği gözlenmiştir. Gecikme duyarlı uygulamalarda ise, enerji duyarlı DCF protokolü EDSMAC protokolüne göre yaklaşık olarak 4 ila 27 kat daha az ortalama uçtan-uca gecikme verdiği gözlenmiştir. EDSMAC protokolü kullanan enerji duyarlı uygulamalar, gecikme duyarlı uygulamalara göre yaklaşık olarak 1,3 ila 1,8 kat daha az enerji tüketmektedirler. Ayrıca gecikme duyarlı uygulamalarda enerji duyarlı DCF ve EDSMAC protokolü enerji tüketim konusunda yaklaşık olarak aynı sonuçları verirken, enerji duyarlı uygulamalarda enerji duyarlı DCF protokolünün EDSMAC protokolüne göre yaklaşık 1,1 ila 1,4 kat daha az enerji tükettiği gözlenmiştir.

COMPARATIVE PERFORMANCE ANALYSIS OF MAC PROTOCOLS FOR WIRELESS SENSOR NETWORKS

Serdar SOLAK

Keywords: Wireless Sensor Network, MAC Protocols, OPNET, TDMA, CSMA, DCF, Delay Sensitive and Energy Aware.

Abstract: In recent years, along with the rapid progresses in communication technologies, the usage area of Wireless Sensor Networks (WSN) has also increased. Their potential applications include military, medical, environmental, commercial, and etc. systems.

One of the main restrictions on the design of WSN is energy consumption. Since the energy consumption ratio of a sensing node affects the life time of WSN, the usage of energy resources efficiently by the sensor nodes has quite important for the WSN, and in this context, many studies are going on in academic and industrial areas. These enduring studies have generally focused on routing and MAC protocols.

The MAC protocols proposed for WSN are divided into two categories as centralized and contention based. While the centralized MAC protocols based on TDMA, the contention based ones generally employ the CSMA/CA protocol.

In this study, TDMA and CSMA/CA protocols which are the underlying techniques for proposed WSN MACs are examined in detail. Furthermore, with the OPNET modeler software, TDMA based EDSMAC protocol and contention based DCF protocol have been modeled and their comparative performance evaluation have also been presented.

According to simulation results obtained, for the energy sensitive applications, the sensing nodes employing DCF protocol have approximately 1.1 to 1.4 times lower power consumption ratio when compared with those nodes employing EDSMAC protocol. Besides, energy sensitive applications, EDSMAC protocol provides 20 to 85 times better average message delay when compared with DCF protocol does.

1. GİRİŞ

Teknolojik alandaki gelişmelerin artmasıyla beraber küçük boyutlarda, az enerji tüketen ve çok fonksiyonlu Algılayıcı Düğümler (AD) geliştirilmektedir. AD'lerin bir araya gelerek kablosuz olarak haberleştiği ortam Kablosuz Algılayıcı Ağlar (KAA) olarak tanımlanır. Son zamanlarda KAA askeri, endüstriyel, ticari ve sağlık gibi bir çok alanda yaygın olarak kullanılmaya başlanmıştır.

KAA tasarımında en önemli problemlerden biri enerji tüketimidir. AD'lerin enerji tüketimi, KAA'nın yaşam sürelerini etkilemektedir. Bu sebepten dolayı AD'lerin enerjilerini verimli kullanmaları KAA için büyük önem taşımaktadır.

Enerjinin daha etkin ve verimli bir şekilde kullanılması için akademik çevrelerde ve sanayide çok sayıda çalışma yapılmaktadır. Bu çalışmalar veri bağı katmanı ve ağ katmanı üzerine yoğunlaşmış durumdadır. Ağ katmanında yapılan çalışmalar veri yönlendirme ile ilgilenirken, veri bağı katmanında yapılan çalışmalar ise çoğunlukla MAC (Medium Access Control, Ortam Erişim Kontrol) protokolleri yani, kablosuz ortama erişim yöntemleri ile ilgilenmektedir.

KAA'da MAC protokolü kullanımı açısından henüz bir standart oluşmadığı için literatürde ortam erişimi konusunda çok sayıda çalışma yapılmaktadır. Bu çalışmaların çoğunda enerji tüketimini en aza indirecek yaklaşımlar geliştirmek birincil amaç iken gecikme konusu ikinci planda yer almaktadır. KAA için önerilen MAC protokolleri genel olarak merkezi (Time Division Multiple Access - TDMA) ve çekişme tabanlı olmak üzere 2'ye ayrılmaktadır [3,10].

Bu tez çalışmasında, TDMA yapısı kullanan enerji ve gecikme duyarlı EDSMAC (Energy Aware and Delay Sensitive MAC) [11] protokolü ile çekişme tabanlı CSMA\CA (Carrier Sense Multiple Access with Collision Avoidance) alt yapısı kullanan standart DCF MAC protokolünün benzetim modelleri OPNET (OPtimized

Network Engineering Tool) Modeler yazılımı kullanılarak yapılmıştır. Ayrıca, standart DCF protokolüne enerji tüketimini azaltmak üzere EDSMAC'de olduğu gibi uyku modu eklenerek, bu iki protokolün enerji ve gecikme başarımlarının karşılaştırılmalı değerlendirmesi de sunulmaktadır.

1.1. Literatürde Yapılan Çalışmalar

KAA'da kullanılan MAC protokollerinin standardı henüz tanımlanmadığı için bu alanda çok sayıda çalışma yapılmaktadır. Ortam erişim protokolleri konusunda literatürde yer alan çalışmalardan bazıları aşağıda yer almaktadır.

2002 yılında Wei Ye, John Heidemann, tarafından KAA'da enerji tüketimini azaltmak üzere SMAC (Sensor MAC) [3] protokolü önerilmiştir. Yapılan çalışmada, AD'ler belirli zamanlarda radyo fonksiyonları kapatılarak uyutulmakta ve belirli periyotlarda çalışmaktadır. Böylece, KAA'ın yaşam süresi açısından son derece önemli olan enerjinin verimli kullanımı sağlanmaktadır.

2002 yılında El Hoidi [4], tarafından yapılan çalışmada KAA için CSMA ve TDMA yapısını beraber kullanan bir MAC protokolü önerilmiştir. Yapılan çalışmada enerji tüketiminin yanı sıra gecikmelerinde indirgenmesini sağlamışlardır.

2003 yılında Tijs van Dam, Koen Langendoen, tarafından yapılan çalışmada TMAC (Timeout MAC) [6] protokolü önerilmiştir. SMAC protokolünde sabit olan iş zamanını uyarlanabilir hale getirmek suretiyle KAA'da enerji tüketiminin azaltılması sağlanırken, gecikmelerin düşürülmesi de hedeflenmiştir.

2003 yılında V. Rajendran, K. Obraczka, J. J. Garcia-Luna-Aceves, yapmış oldukları çalışmada enerji kullanımında verimi arttırmak amacıyla TDMA tabanlı TRAMA [7] protokolünü önermişlerdir. CSMA tabanlı protokollere göre AD'nin uykuda kalma süresi çok yüksek olmasına rağmen çarpışma olasılıklarını düşük seviyede tutmaktadır.

2004 yılında C.C. Enz, A. El-Hoiydi, J-D. Decotignie, V. Peiris, tarafından daha önceden El Hoidi tarafından yapılan çalışma geliştirilerek WiseMAC [5] protokolü önerilmiştir. İletim ortamında kontrol kanalına erişmek için CSMA kullanılırken, veri kanalına erişim için TDMA kullanılmaktadır. Enerji ve gecikmelere karşı duyarlı bir yaklaşım önermişlerdir.

2004 yılında P. Lin, C. Qiao ve X. Wang tarafından yapılan çalışmada SMAC protokolü geliştirilerek DSMAC (Dynamic Sensor MAC) [8] protokolü önerilmiştir. TMAC protokolüne benzer şekilde çalışmaktadır. SMAC protokolünde sabit olan iş zamanı uyarlanabilir hale getirilmek suretiyle enerji tüketiminin yanı sıra gecikmelerinde indirgenmesi sağlanmaktadır.

2004 yılında Wei Ye, John Heidemann, 2002 yılında önermiş oldukları SMAC protokolüne uyarlanabilir uyku modu eklemişlerdir. Yapılan çalışma sonunda enerji tüketimini indirmenin yanı sıra gecikmeleri de indirgemişlerdir [9].

2004 yılında J. Polastre, J. Hill, D. Culler tarafından yapılan çalışmada BMAC (Berkeley MAC) [15] protokolü önerilmiştir. IEEE 802.11 DCF protokolünden esinlenerek tasarlanan protokol yüksek kanal kullanımı, düşük seviyede güç tüketim işlemi ve verimli çarpışmadan kaçınma özellikleri sayesinde SMAC protokolünden daha iyi sonuçlar üretmiştir.

2005 yılında P.C. Nar, E. Cayirci yapmış oldukları çalışmada PCSMAC (Power Controlled Sensor-MAC)[10] protokolünü önermişlerdir. Bu çalışmada SMAC protokolüne güç kontrol kabiliyeti eklenmiştir. RTS (Request To Send), CTS (Clear To Send), veri ve ACK (ACKnowledgement) paketlerinin iletimini en uygun zamanda gerçekleştirmektedir. Bu sayede enerji tüketimi başarımı SMAC protokolüne göre 3 kat daha iyi sonuç vermiştir [10].

2006 yılında Michael I. Brownfield, Kaveh Mehrjoo, Almohonad S. Fayez, Nathaniel J. Davis, yapmış oldukları çalışmada GMAC (Gateway MAC) [14] protokolünü önermişlerdir. AD'lerin uyku sürelerini maksimum seviyede tutup, bekleme seviyesini minimuma indirgemişler ve bu sayede KAA'nın yaşam süresini

arttırmışlardır. Ayrıca önerilen MAC protokolünü, SMAC, TMAC ve IEEE 802.11 protokolleri ile karşılaştırarak başarım analizi yapmışlardır.
2006 yılında I. Demirkol, C. Ersoy, F. Alagöz [13] yapmış oldukları çalışmada KAA için önerilen SMAC, TMAC, DSMAC ve WiseMAC protokolleri hakkında genel bilgiler vermişler ve bu protokolleri karşılaştırmışlardır.

2007 yılında C. Çeken yapmış olduğu çalışmada enerji ve gecikmelere karşı duyarlı EDSMAC [11] protokolünü önermiştir. Yeni MAC protokolünde kullanılan algoritmalar ile kablosuz algılayıcı düğümün mümkün olduğu kadar az enerji tüketmesi sağlanmıştır. Mevcut slotlar içerisine yerleştirilen çoğuşma (burst) slotları ile düşük güç tüketiminin yanı sıra uçtan-uca mesaj gecikmesi de azaltılmış ve özellikle gerçek zamanlı uygulamalara servis kalitesi (QoS) desteği sağlanmıştır.

1.2. Tez Çalışmasının Amacı

Bu tez çalışmasının temel amacı KAA için önerilmiş olan MAC protokollerini incelemek ve değişik trafik yükleri altında enerji tüketimi ve gecikme duyarlılığı bakımından başarımlarını karşılaştırmaktır. Bu doğrultuda aşağıdaki çalışmalar gerçekleştirilmiştir:

- KAA çalışması, kullanım alanları, protokol yapısı ve enerji tüketim kaynaklarının incelenmesi.
- KAA'da TDMA ve çekişme tabanlı olarak önerilen MAC protokollerin incelenmesi.
- Benzetim çalışmasında kullanılacak OPNET Modeler yazılımının öğrenilmesi.
- TDMA tabanlı EDSMAC protokolünün başarım sonuçlarının incelenmesi.
- Çekişme tabanlı enerji duyarlı DCF protokolünün benzetimi ve sonuçlarının incelenmesi.
- Benzetim sonuçlarının karşılaştırmalı olarak değerlendirilmesi.

1.3. Tez Çalışmasının Katkıları

Yukarıda verilen amaç ve hedefler doğrultusunda yapılan tez çalışmasının temel katkıları özetle şunlardır:

- KAA hakkında temel bilgiler detaylı olarak sunulmaktadır.
- TDMA ve CSMA tabanlı MAC protokollerinin çalışması hakkında detaylı bilgiler sunulmaktadır.
- Literatürde yer alan KAA MAC protokolleri hakkında bilgiler sunulmaktadır.
- TDMA alt yapısını kullanan enerji ve gecikmelere karşı duyarlı EDSMAC protokolünün detaylı olarak anlatımı ve benzetimi, modeli sunulmaktadır.
- CSMA alt yapısını kullanan enerji duyarlı DCF MAC protokolünün enerji ve gecikme duyarlı hali detaylı olarak anlatılmakta ve benzetimi sunulmaktadır.
- Benzetimleri yapılmış olan MAC protokollerinin enerji ve gecikme başarımları karşılaştırmalı olarak değerlendirilmektedir.

1.4. Tez Organizasyonu

Tez çalışması altı ana başlıktan oluşmaktadır ve aşağıdaki şekilde organize edilmiştir.

Bölüm 2'de, KAA'ın tanımı, kullanımı, çalışma mantığı, AD düğüm modeli, protokol yapısı ve enerji probleminin sebepleri hakkında bilgi verilmektedir.

Bölüm 3'de, KAA'da önerilen MAC protokollerinin alt yapısını oluşturan TDMA ve çekişme tabanlı yöntemler hakkında detaylı bilgiler verilmektedir.

Bölüm 4'te, OPNET Modeler yazılımı, TDMA alt yapısını kullanan enerji ve gecikme duyarlı EDSMAC protokolü, CSMA\CA alt yapısını kullanan enerji duyarlı DCF MAC protokolü hakkında detaylı bilgiler ve oluşturulan düğüm ve proses modelleri bulunmaktadır.

Bölüm 5’te, OPNET modeli gerçekleştirilen iki protokolün yer aldığı örnek bir KAA senaryosu ve bu senaryonun benzetiminden elde edilen sonuçların karşılaştırmalı başarım değerlendirmesi sunulmaktadır.

Bölüm 6’da, tez kapsamında yapılan çalışmalardan elde edilen sonuçlar genel hatlarıyla değerlendirilmektedir.

2. KABLOSUZ ALGILAYICI AĞLAR

2.1. Giriş

Son zamanlarda hızla gelişen iletişim teknolojisiyle beraber, düşük maliyet ve güçte, kısa mesafelerde birbiriyle haberleşebilen çok işlevli cihazlar geliştirilmektedir. Algılama, hesaplama ve iletişim yeteneklerine sahip bu küçük cihazlara algılayıcı (sensör) denilmektedir [24]. AD'lerin verileri işlemek, çevreyi gözetlemek ve fiziksel dünya ile iletişimi sağlamak amacıyla kablosuz olarak haberleştirildiği ortama Kablosuz Algılayıcı Ağlar denilmektedir.

KAA yapısı düzensiz bir yapı olup farklı özelliklere sahip bir çok AD'den meydana gelmektedir. AD'lerin kapasite ve boyut gibi farklılıklarının yanı sıra algılamış oldukları veriye göre de farklılık gösterirler. AD fiziksel ortamdan sıcaklık, ses, görüntü, basınç, hız, yön, hareket, miktar, aydınlık ve nem [1,2] gibi çeşitli bilgileri algılamak ve bu bilgileri kablosuz ortam üzerinden merkezi düğüme (MD) iletmek üzere kullanılmaktadır.

2.2. Kablosuz Algılayıcı Ağların Kullanım Alanları

KAA'ın kullanım alanları her geçen gün artmaktadır. Askeri, çevre, sağlık, ticari vb. alanlarda yaygın olarak kullanılmaktadır. Bunlardan bazıları aşağıda belirtilmiştir [1,23,27,30].

2.2.1. Askeri uygulamalar

Savaş alanlarının izlenmesi, düşman hareketlerinin izlenmesi, arazi hakkında keşifte bulunmak, personel ve askeri araçların takip edilmesi, dost kuvvetlerin izlenmesi ve hedeflerin hız ve konumlarının tespit edilmesinde kullanılmaktadır.

2.2.2. Çevresel uygulamalar

Hava durumu, hava kirliliğinin tespiti, sel, deprem, orman yangını gibi doğal afetlerin takip edilmesi, tarımsal faaliyetlerin izlenmesi gibi uygulamalarda kullanılır.

2.2.3. Sağlık uygulamaları

Hastanede bulunan doktorların yerinin tespit edilmesi, hastaların durumlarının takip edilmesi ve çeşitli sağlıksal parametrelerin takip edilmesinde kullanılır.

2.2.4. Ticari uygulamalar

Araçların izlenmesi ve tespit edilmesi, enerji hatlarının izlenmesi, küçük çocukların aileleri tarafından takip edilmesi, ışıklandırma kontrolü, trafik ışıklarının kontrolü, yangın sistemleri gibi alanlarda kullanılır.

2.3. Kablosuz Algılayıcı Ağlarda Enerji Probleminin Sebepleri

KAA'nın en büyük problemi enerji kullanımıdır. Enerji kullanımını etkin bir hale getirmek için enerjinin nerelerde harcandığının ve nerelerde gereksiz yere harcandığının bilinmesi gerekir [2,13].

2.3.1. Veri iletimi

AD'lerin verilerini iletmesi veya yönlendirmesi gereken paketlerin iletilmesi enerji sarfiyatına sebep olmaktadır. Araştırmalar 1 bit verinin A noktasından B noktasına iletilmesinin 1000 adet mikro kod yürütülmesine eşdeğer enerji tüketildiğini göstermektedir. Verinin sıkıştırılarak gönderilmesi veya paketlerin birleştirilerek gönderilmesi enerji tüketimini azaltacaktır.

2.3.2. Yönlendirme

KAA'da yönlendirme işlemi çıkış düğümüne (sink) yakın olan AD'ye doğru olmaktadır. Bu sebepten dolayı çıkış düğümüne yakın olan AD'ler diğer AD'lerin paketlerini de çıkış düğümüne ileteceğinden bu AD'de enerji sarfiyatı diğer AD'lere göre daha fazla olacaktır.

2.3.3. Çarpışma (Collision)

Ortak haberleşme yolunun kullanıldığı AD'lerde komşu sayısı da fazla ise çarpışma olasılığı ve paketlerin tekrar gönderilme sıklığı artacağından enerji sarfiyatı da artacaktır. Bazı AD'lerin bazı yolları kullanmamalarını sağlamak çarpışma olasılığını azaltacaktır.

2.3.4. Dinleme

AD'ler iletim yapmadıkları zaman uykuya veya dinlemeye geçerler. AD'ler uyuma sırasında nerdeyse hiç enerji harcamazken, dinleme sırasında iletim yapar gibi enerji harcamaktadırlar.

2.3.5. Hata oranı

Paketlerin iletilmesi sırasında hata oluşmasından dolayı paketler tekrar gönderilir. Bu tekrarlardan dolayı enerji tüketimi artmaktadır.

2.3.6. Zayıf düğüm

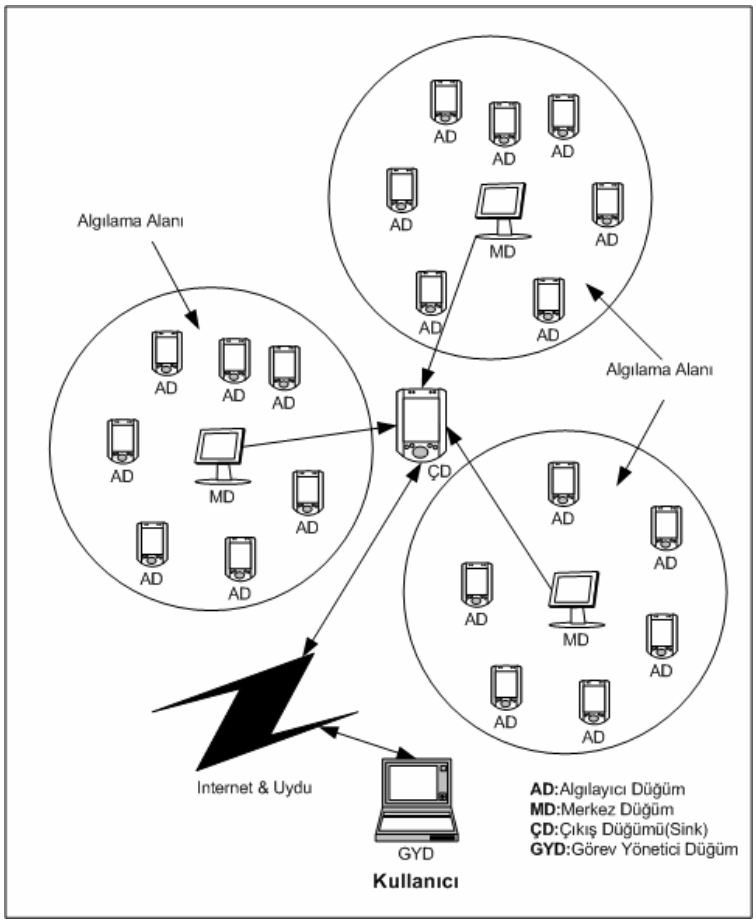
Zaman ilerledikçe KAA içerisinde enerji bakımından diğer AD'lerden daha zayıf AD'ler oluşmaktadır. Bu AD'lerin belirlenerek, yönlendirme için kullanılmaması KAA'nın yaşam süresini arttırmaktadır.

2.3.7. Yük paylaşımı

AD'lerden çıkış düğümüne doğru giden yolların fazla olması ve yüklerin adil bir şekilde dağıtılması KAA'nın yaşam süresini arttırmaktadır.

2.4. Kablosuz Algılayıcı Ağların Çalışması

KAA'da bulunan AD'ler temel olarak algılama, veri işleme ve işlenen verileri diğer AD'lere iletme işlemlerini gerçekleştirirler. Fiziksel dünyadan algılamış oldukları verileri işleyerek sanal dünyaya taşımaktadırlar. Şekil 2.1'de KAA' nın çalışmasını gösteren bir örnek bulunmaktadır.

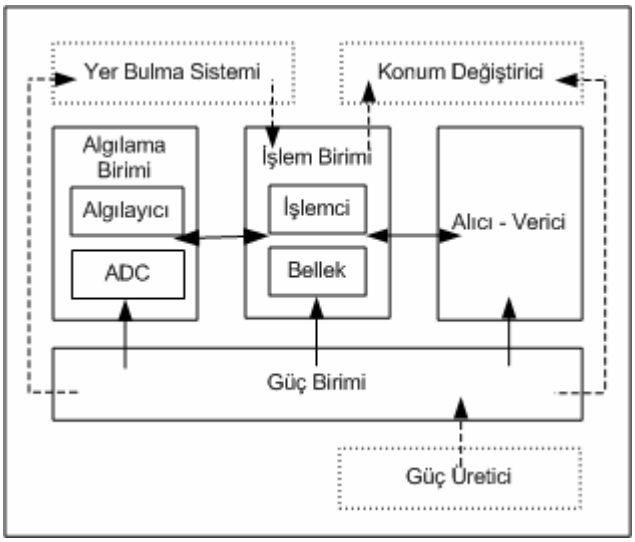


Şekil 2.1: KAA'nın çalışmasını gösteren örnek

AD'ler algılama alanına düzensiz bir yapıda saçılmaktadır. AD'ler fiziksel ortamdan algılamış oldukları verileri işleyerek komşu düğüme veya MD' ye iletilmektedir. MD kendine gelen verileri işleyerek çıkış düğümüne göndermektedir. Çıkış düğümü de internet veya uydu yardımıyla verileri görev yönetici düğüme ulaştırmaktadır [1].

2.5. Algılayıcı Düğümün Yapısı

AD'lerin genel yapısı algılama, işlem, alıcı/verici ve güç birimlerinden meydana gelmektedir. Bazı durumlarda bunlara ek olarak algılayıcıda, yer bulma sistemi, konum değiştirici ve güç üreteci bulunmaktadır [1]. Şekil 2.2'de AD'nin genel yapısı görülmektedir.

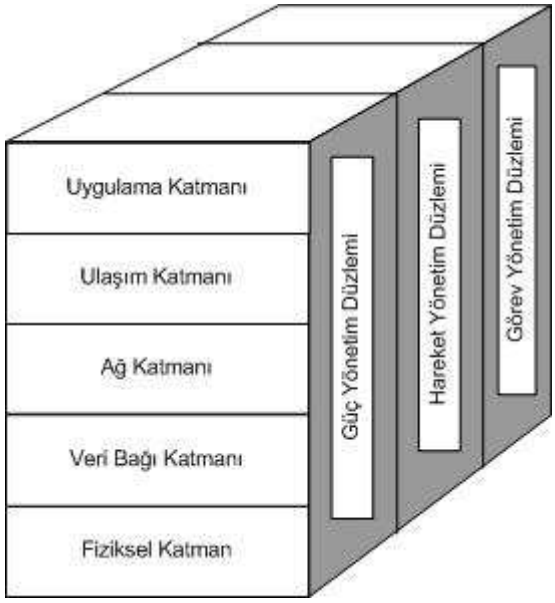


Şekil 2.2: AD'nin genel yapısı

Algılama birimi, nem, basınç, sıcaklık gibi fiziksel ölçüleri algılayıp işlem biriminin anlayacağı sayısal sinyallere çevirmektedir. İşlem birimi, AD'yi yöneten kısımdır. Alıcı – Verici birimi, AD ile KAA arasında iletişimi sağlar. Güç birimi , AD'nin en önemli kısmıdır. AD' de bulunan diğer birimlerin güç ihtiyacını karşılamak için kullanılmaktadır. Güç biriminin daha fazla dayanması veya AD' nin hayatını daha fazla sürdürmesi için bazen güç üreteci eklenmektedir. Bazı uygulamalarda algılanan verilerin işlenmesi için yol bulma birimine ihtiyaç vardır. Önerilen yönlendirme protokolleri AD'lerde yol bulma sisteminin bir ihtiyaç olduğunu göstermektedir. Ayrıca hareket eden AD'ler için konum değiştirici kullanılmaktadır [1,23].

2.6. Kablosuz Algılayıcı Ağların Protokol Yapısı

KAA'da kullanılan protokol yapısı Şekil 2.3'te görülmektedir. Protokol yapısı, güç ve yol bulma gibi özellikleri de içinde barındırmaktadır.



Şekil 2.3: KAA'da kullanılan protokol yapısı

Protokol yapısı, fiziksel katman, veri bağı katmanı, ağ katmanı, ulaşım katmanı, uygulama katmanı, güç yönetim düzlemi, hareket yönetim düzlemi ve görev yönetim düzlemi olarak bölümlendirilmiştir [1].

Fiziksel katman, frekans seçimi, modülasyon, veri şifreleme ve iletilmesinden sorumlu olan katmandır. Veri bağı katmanı, ortama erişim kontrolü (MAC), hata kontrolü, veri akışının çoklanması olaylarından sorumlu olan kısımdır. Kablosuz ortamın gürültülü ve AD'lerin hareketli olmasından dolayı, MAC protokolünün güç yönetimi ve komşularda yapılan yayınlara karşı çarpışmaları minimize etme özelliği taşıması gerekir. Ağ katmanı, ulaşım katmanı tarafından iletilen verinin yol bulma işlemiyle ilgilenmektedir.Ulaşım katmanı, veri akış kontrolü yapar. Algılama işlemleri sayesinde farklı tipte uygulamalar geliştirilebilir.Bu uygulamalar, uygulama katmanı sayesinde kullanılmaktadır [28].

Bu katmanların yanı sıra protokol yapısında, güç, hareket ve görev yönetim düzlemleri bulunmaktadır. Bu düzlemler, AD'lerin algılama işlemlerini düzenlemek ve enerji tüketimini düşük seviyede tutmaya yardımcı olmaktadırlar.

Güç yönetim düzlemi, AD'nin güç kullanımını yönetmektedir. AD, herhangi bir komşu AD'den paketi aldıktan sonra kapatılarak aynı paketi tekrar alması önlenebilir. AD'nin enerji seviyesi alt eşik değerinin altına düşerse, komşu AD'lere mesaj göndererek yol bulma işlemlerine katılamayacağı bildirebilir. Bu sayede AD kalan enerjisini sadece algılama işlemi için kullanabilir.

Hareket yönetim düzlemi, AD'lerin hareketlerini algılayarak kaydetmektedir. Bu sayede komşu AD'leri ve geri dönüş yolunu rahatlıkla bulunmaktadır. Komşu AD'lerin bilinmesi, AD'nin görev ve güç yönetimini de dengelemektedir.

Görev yönetim düzlemi, AD'lerin algılama alanda yapacakları algılama görevlerini paylaştırmaktadır.

Yönetim düzlemleri, AD'lerin güçlerini verimli bir şekilde kullanmalarını, hareketli KAA'da verinin yol bulmasını ve AD'ler arasında kaynakların paylaşımını sağladıklarından KAA için büyük bir önem taşımaktadır.

2.7. Kablosuz Algılayıcı Ağların Avantaj ve Dezavantajları

2.7.1. Avantajları

KAA'nın kullanılması ile ilgili avantajları özetleyecek olursak;

- Hatalara karşı toleranslı olması : KAA'da çok sayıda AD bulunmaktadır. Bu AD'lerin, sınırlı güç, fiziksel hasar ve çevresel sebeplerden dolayı hata verip işlemleri durabilir. Bu durumda KAA'nın performansının fazla etkilenmeden işlemlerine devam etmesi gerekmektedir. Bu durum KAA'nın hatalara karşı toleranslı olmasından kaynaklanmaktadır.

- Ölçeklenebilirlik (Scalability) : İzleme yapılacak olan bölgede çok sayıda AD bulunabilir. Yeni AD'lerin algılama alanına katılması mümkündür. Yeni KAA'nın, izleme bölgesinde bulunan KAA'ya katılması mümkün olmaktadır. KAA'nın bu katılımlardan sonrada işlevine devam etmesi ölçeklenebilirlikle alakalıdır.
- Üretim maliyeti : KAA çok sayıda AD'den meydana geldiğinden tek bir AD'nin maliyeti tüm KAA'nın maliyeti için önem taşımaktadır. Gelişen mikroişlemci teknolojisi sayesinde AD'lerin maliyeti ucuzlamıştır.
- Kullanım kolaylığı : AD'ler herhangi bir ayar gerektirmeden kendi aralarında organize olarak dinamik bir şekilde çalışmakta ve değişen koşullara ayak uydurabilmektedirler.
- Yeniden kullanılması : AD'ler fiziksel ortamdan çeşitli olayları algılamak için kullanılmaktadır. Bu sebepten dolayı farklı uygulamalarda ilgili olayları algılamak için yeniden kullanılmaktadırlar.
- Taşınabilir olması : Kablo ve enerji alt yapısı gerektirmediğinden KAA'nın bir yerden başka bir yere kolaylıkla taşınması gerçekleştirilmektedir.
- Algılayıcı düğümlerin hareketi : Algılama alanında kablosuz olarak haberleşen AD'ler herhangi bir kısıtlama olmadan hareket edebilmektedirler.

2.7.2. Dezavantajları

KAA'nın sağladığı avantajların yanı sıra dezavantajları da bulunmaktadır. Kaynakların kısıtlı olması, AD'lerin izlenmesi ve yönetilmesi ile ilgili zorluklar, hata olasılığının yüksek olması ve servis kalitesinin istenen seviyede olmaması gibi dezavantajları bulunmaktadır.

En önemli dezavantajı kaynakların kısıtlı olmasıdır. Enerji ve bellek, AD için önemli bir yer teşkil etmektedir. Belleğinin kısıtlı olması sebebiyle ileri derece algoritmik işlemlerin yapılması zor olmaktadır. Enerjinin sınırlı olması sebebiyle AD'nin hayatta kalma süresi de sınırlı olmaktadır.

2.8. Sonuç

Son zamanlarda hızla gelişen iletişim teknolojisiyle beraber, KAA'nın kullanım alanları da artmaktadır. Özellikle, sel, deprem, yangın gibi çevresel faaliyetlerin izlenmesi, dost ve düşman kuvvetlerin takip edilmesi gibi askeri alanlarda ve hastanelerde doktorların izlenmesinde yaygın olarak kullanılmaya başlanmıştır.

Mikroişlemci teknolojisinin gelişmesiyle AD'lerin maliyetlerinin ucuzlaması KAA genel maliyetini düşürdüğünden kullanım alanlarının hızla artma sebeplerinden biridir. Bunun yanı sıra KAA'nın en büyük problemlerinden biri enerji tüketimidir. Gerek akademik gerek endüstriyel alanda yapılan çalışmaların odak noktası enerji kullanımını verimli hale getirmek olmuştur. Enerji kullanımını verimli hale getirebilme adına en önemli görev kullanılacak ortam erişim mekanizmasına düşmektedir.

3. KABLOSUZ ALGILAYICI AĞ MAC PROTOKOLLERİ

3.1. Giriş

Ortama erişim kontrolü (MAC), kablosuz iletim ortamının düğümler arasında etkin bir şekilde paylaştırılmasını sağlayan bir mekanizmadır. MAC katmanı, veri paketlerinin parçalanması, hata iyileştirme, hareket yönetimi, güç koruma ve şifreleme gibi işlemleri kapsamaktadır.

KAA'da MAC katmanı üzerinde yapılan çalışmalar enerji ve gecikmeleri azaltmak üzere yoğunlaşmış durumdadır. KAA'da henüz standart bir MAC protokolü oluşturulamamakla birlikte bu konuda literatürde yapılan çok sayıda çalışma bulunmaktadır.

KAA'da enerji kullanımının büyük bölümü radyo iletişim fonksiyonları tarafından gerçekleştirilmektedir. [12]'deki çalışmada AD'lerde radyo fonksiyonlarının kullanım oranları, bekleme, veri alma ve veri göndermede, deneysel olarak, sırasıyla 1, 1,05 ve 1,4 olarak gözlenmiştir.

KAA'da MAC protokolleri, TDMA (merkezi) ve çekişme tabanlı olmak üzere ikiye ayrılmaktadır [3,10,29]. Çekişme tabanlı MAC protokolleri genellikle DCF (Distributed Coordination Function) [16] yapısını kullanmaktadır. KAA'da önerilen MAC protokolleri bu iki yapıdan birini veya iki yapının karışımını kullanmaktadır.

KAA'da enerji tüketimini indirgemek için SMAC (Sensor MAC) [3,9], TMAC (Timeout MAC) [6], DSMAC (Dynamic Sensor MAC) [8], WiseMAC [4,5],gibi birçok MAC protokolü önerilmiştir.

SMAC protokolü KAA'da enerji tüketimini indirgemek amacıyla önerilen bir MAC protokolüdür. CSMA/CA (DCF) altyapısı kullanan SMAC protokolü gecikme

duyarlılığını göz önünde bulundurmazken, enerji tasarrufu sağlamak üzere AD' ler belirli periyotlarla uyku moduna alınmaktadır [3,9,13].

TMAC [6] ve DSMAC [8] protokolü, SMAC protokolünün değişken trafik yükleri altında oluşan kötü sonuçlarını iyileştirmek için önerilen bir protokoldür. SMAC'te sabit olan uyku ve çalıma periyotları bu protokolde değişken hale getirilerek hem enerji tüketiminde hem de gecikmelerde verimlilik sağlanmaktadır [13].

WiseMAC protokolü TDMA ve CSMA protokollerinin birleştirilmesi sonucunda oluşturulmuş değişken trafik yükleri altında enerji tüketimini indirgemek için önerilen bir MAC protokolüdür [5].

KAA için geliştirilen MAC protokollerinin temelini oluşturan, TDMA ve DCF protokolleri takip eden alt bölümlerde detaylı bir şekilde anlatılmaktadır.

3.2. DCF Protokolü

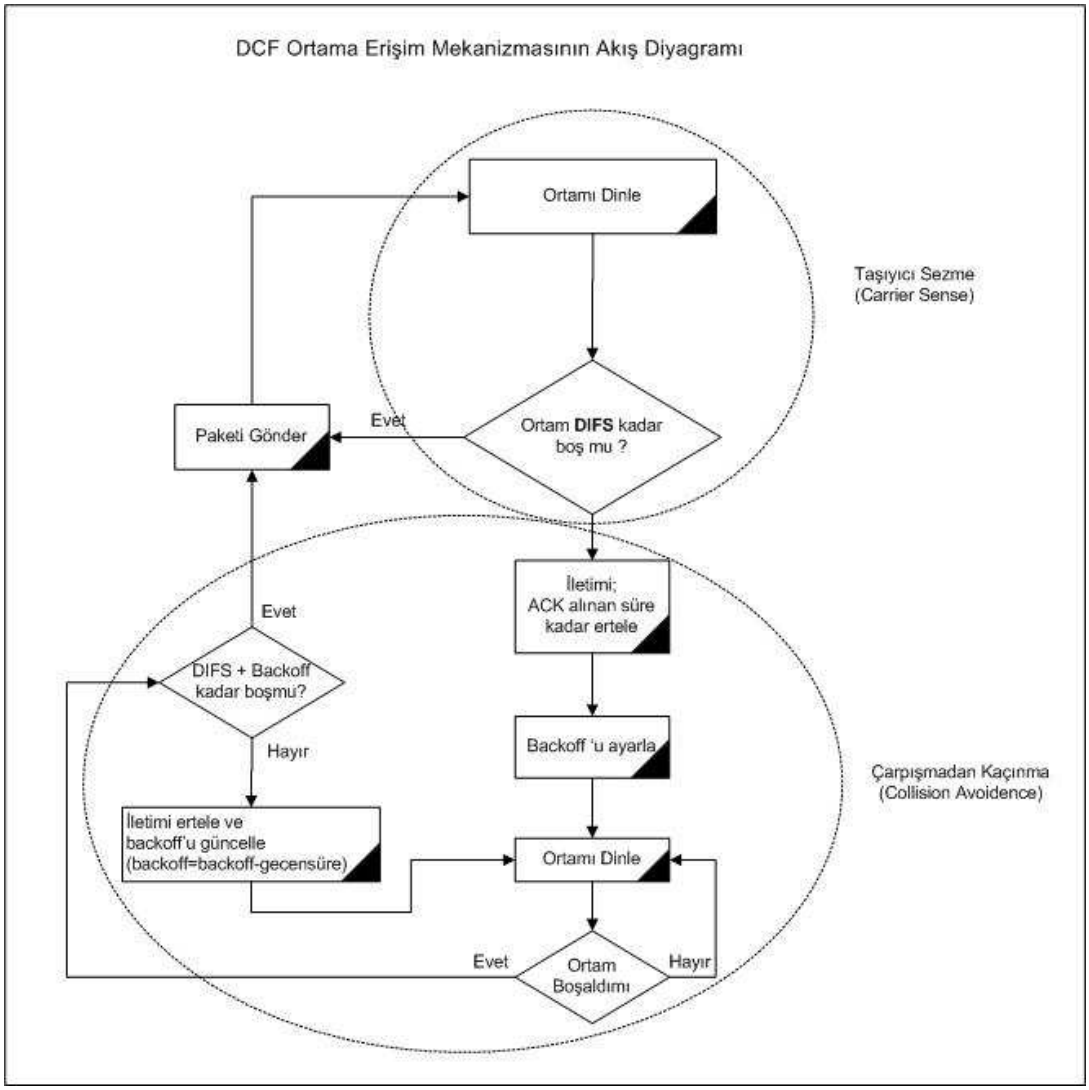
KAA'da çekişme tabanlı MAC protokolleri DCF [16,17] alt yapısını kullanarak tasarlanmaktadır. IEEE 802.11 DCF, IEEE 802.11 standardında ortama erişmek için kullanılan CSMA\CA'nın dağıtık eşgüdümlü fonksiyonudur. Çalışma prensibi "konuşmadan önce dinle" mantığına dayanmaktadır. İletim ortamının kullanımını belirlemek için taşıyıcı sezme (Carrier Sense – CS) ve ortamdaki çarpışmayı (collision) çözmek içinde Backoff algoritması kullanılır.

Çarpışmadan kaçınma (Collision Avoidance), iletim yapmak isteyen AD'lerin farklı zaman aralıklarında iletim ortamına erişmeleri için kullanılmaktadır. AD'lerin farklı zaman aralıklarında ortama erişmeleri, rasgele bir zaman aralığında (Backoff zamanı) erişim demektir. Backoff zamanı, çerçeveler arasında bulunan zamanın (slot time) rasgele bir katıdır. Burada bahsedilen rasgele değer 0-CW (Contention Window) (CWmin, CWmax) arasındadır. Backoff zamanı, 0-CW arasındaki rasgele değerle zaman diliminin çarpılması sonucu ortaya çıkmaktadır. CWmin değeri 31, CWmax değeri 1023 tür. CW değerine ilk olarak CWmin değeri atanır. İletimin tekrar gerektiği durumlarda CW değeri 2 katına çıkarılır. CW değerinin alacağı üst

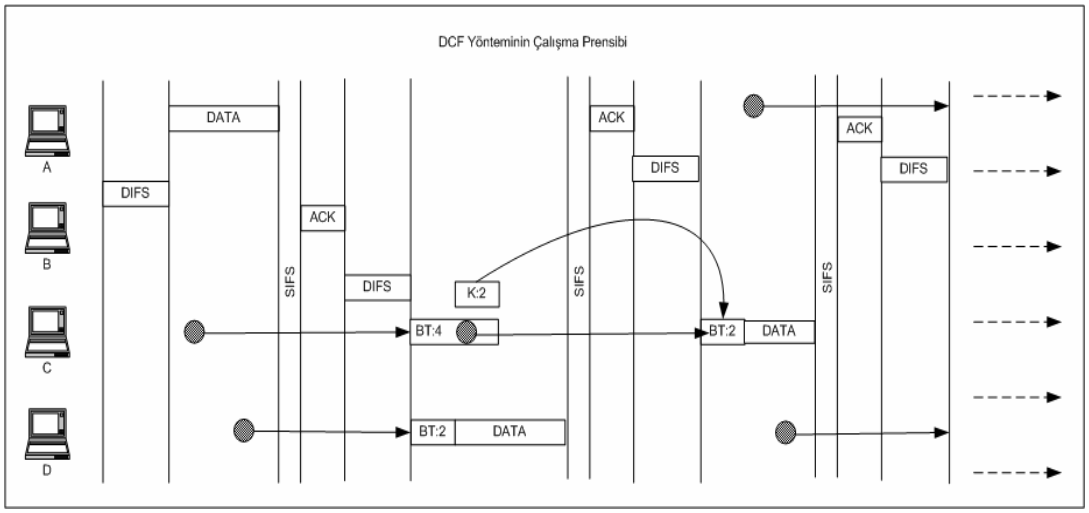
eşik değeri CWmax değeridir. Başarılı iletim yapıldığı zaman CW değerine CWmin değeri atanmaktadır [16,17,19,20].

IEEE 802.11 DCF yönteminde ortam ilk olarak DIFS (Distributed Coordination Function IFS) kadar dinlenir. Ortam meşgul değilse iletim yapılır. Eğer ortam meşgulse, Backoff algoritması uygulanarak, 0 – CW arasında rasgele bir sayı seçilir ve iletim ertelenir. Ortam boşaldıktan sonra, DIFS kadar dinlenir ve rasgele seçilmiş olan Backoff zamanı kadar beklenir. Ortam hala boşsa iletim yapılır. Eğer ortam meşgulse Backoff değeri güncellenir ve iletim ertelenir. Bu işlem, Backoff değeri sıfır (‘0’) olup iletim yapılana kadar devam etmektedir. Eğer iletim ortamında çarpışma olursa CW değeri 2 katına çıkmaktadır. Başarılı iletimden sonra tekrar CW değeri CWmin değerine indirgenmektedir. Şekil 3.1’de DCF ortama erişim mekanizmasının akış şeması bulunmaktadır [20,30].

Şekil 3.2’de DCF ortama erişim mekanizmasının çalışmasını gösteren bir örnek bulunmaktadır. A, B, C ve D düğümleri iletimde bulunmak istemektedirler. A düğümü DIFS kadar bekledikten sonra ortamın meşgul olmadığını anlamış ve paketini göndermiştir. A düğümü paketini gönderdikten sonra C ve D düğümleri de paketlerini göndermek istemişler fakat ortamın meşgul olduğunu anlamışlardır. Bu durumda C ve D düğümlerine Backoff algoritması uygulanmaya başlanmıştır. Rasgele değerler seçilmiştir. C düğümü için 4 ve D düğümü için 2 değerleri seçilmiştir. Ortam boşaldıktan sonra DIFS kadar beklenmekte ve arkasından Backoff rasgele değeri ile zaman dilimi çarpılarak elde edilen Backoff süresi kadar beklenmektedir. Bu süre sonunda, D düğümü daha küçük bir Backoff değeri seçtiğinden iletim yapmaktadır. C düğümü ise ortamın meşgul olduğunu anlayıp, Backoff değerini güncelleyerek iletimini ertelemektedir.



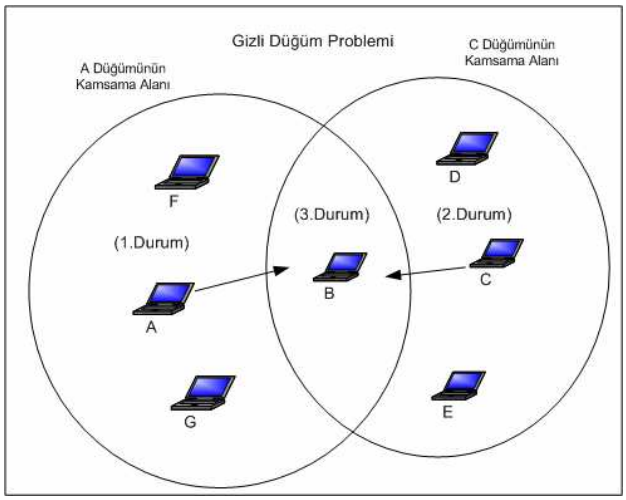
Şekil 3.1: DCF ortama erişim mekanizmasının akış şeması



Şekil 3.2: DCF ortama erişim mekanizmasının çalışmasını gösteren örnek

3.2.1. Gizli düğüm problemi (Hidden node problem)

Gizli düğüm problemi, kablosuz ortamda kapsama alanından kaynaklanan bir problemdir [20]. İletim ortamında 3 düğüm olduğu varsayılırsa; A düğümü, B düğümü ile iletimde bulunmak isterken aynı anda A düğümünün kapsama alanı dışında olan C düğümü de B düğümü ile iletimde bulunmak isteyebilir.(B düğümü A ve C düğümlerinin kapsama alanındadır.) Bu durumda çarpışma olacaktır. Şekil 3.3'de gizli düğüm problemini gösteren resim bulunmaktadır.



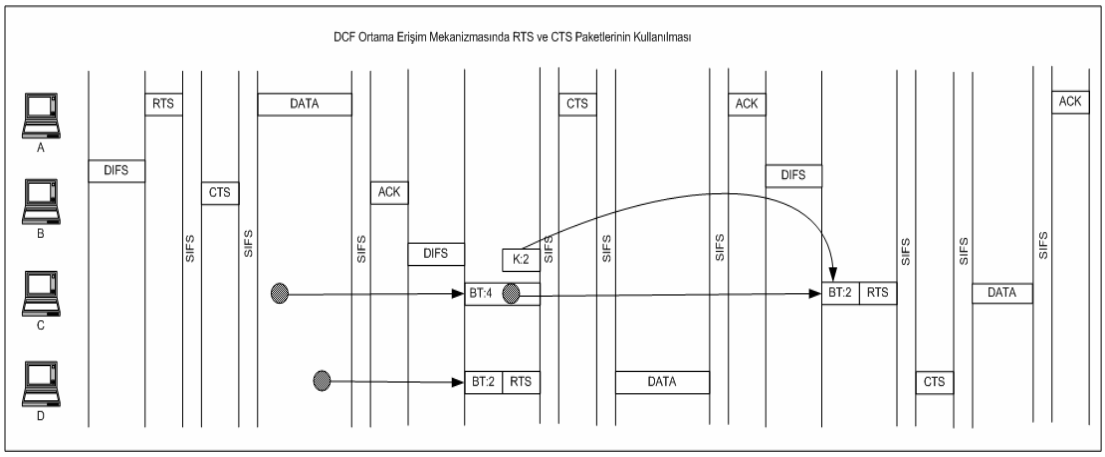
Şekil 3.3: Gizli düğüm problemi

Şekil 3.3'de gizli düğüm probleminin oluşması ile ilgili olarak 3 durum söz konusudur.

- 1.Durum; A düğümü, B düğümüne paket yollamak istiyor. Ortamı dinliyor ve uygun olduğunu anlayıp paketini gönderiyor.
- 2.Durum; C düğümü, B düğümüne paket yollamak istiyor. Ortamı dinliyor ve uygun olduğunu anlayıp paketini gönderiyor.
- 3.Durum; B düğümü A ve C düğümlerinin kapsama alanında olduğundan ve iki düğümde paketlerini gönderdiğinden çarpışma olmaktadır.

3.2.2. DCF protokolünde gizli düğüm probleminin çözümü

DCF ortama erişim mekanizmasında gizli düğüm problemini önlemek için RTS (Request To Send) ve CTS (Clear To Send) paketleri kullanılmaktadır. [20] İletim yapılmadan önce RTS paketi gönderilir ve alıcıdan CTS paketi beklenir. CTS paketi geldikten sonra iletim yapılmaya başlanmaktadır. Eğer CTS paketi gelmezse, bir süre sonra alıcıya yeniden RTS paketi göndermektedir. Şekil 3.4'te DCF ortama erişim mekanizmasının RTS ve CTS paketlerini kullanarak çalışması görülmektedir.

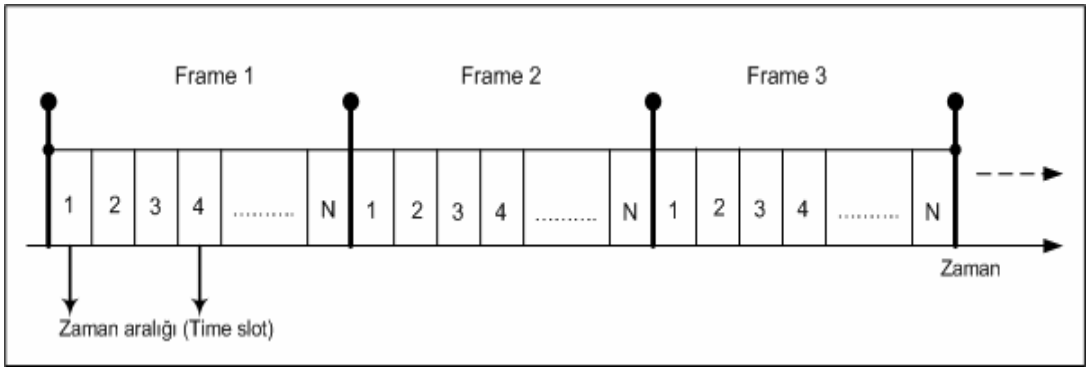


Şekil 3.4: DCF yönteminin RTS ve CTS paketleriyle beraber kullanılması

3.3. TDMA Protokolü

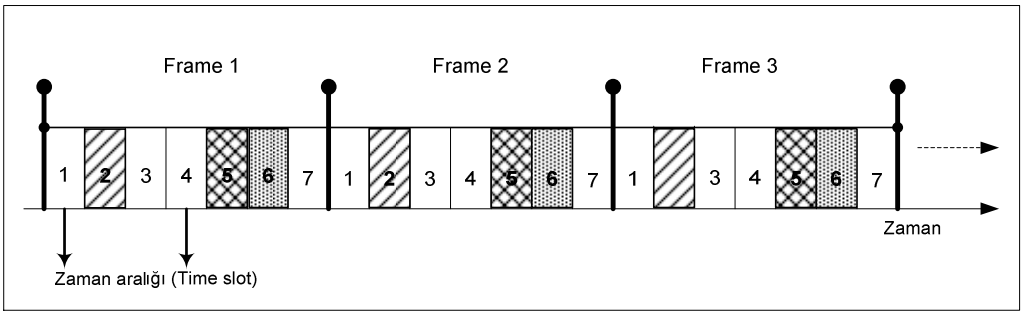
KAA'da merkez tabanlı MAC protokolleri TDMA [18,21,25] alt yapısı kullanılarak tasarlanmaktadır. TDMA yönteminde, iletim zamanı her biri eşit küçük zaman aralıklarına (time slot) bölünmektedir. Her düğüm, kendine ayrılmış olan zaman aralığında iletimini gerçekleştirmektedir. Her düğüm kendi zaman aralığında iletimini gerçekleştirdiğinden sürekli iletişim yapılmamaktadır. İletim zamanı içerisinde kullanılmayan zaman aralıkları boş kalmaktadır. Veri iletiminin sürekli olmadığı patlamalı trafiklerde enerji tasarrufu sağlamaktadır. Düğümler aynı bant genişliğini kullandığından senkronizasyonunun dikkatli yapılması gerekmektedir.

Şekil 3.5'te TDMA protokolüne göre iletim zamanları (Frame), n adet küçük zaman aralıklarına bölünmüştür [21,25].



Şekil 3.5: N adet düğüm için TDMA protokolü

Şekil 3.6'da TDMA protokolünün çalışmasına bir örnek verilmektedir. İletim zamanı 7 slota bölünmüştür. Her düğüm kendine ayrılan slot zamanında iletim yapmaktadır. 2, 5 ve 6 numaralı düğümler iletimlerini yaparken, 1, 3, 4 ve 7 numaralı düğümlerin gönderecek paketleri bulunmadığından iletim yapmamaktadırlar. Klasik TDMA yönteminde iletim yapmayan düğümlerin slotları diğer düğümler tarafından kullanılmamaktadır. Bu ise mevcut bant genişliğinin etkin kullanılamaması anlamına gelmektedir.



Şekil 3.6: TDMA protokolünün çalışmasına örnek

3.4. Sonuç

KAA MAC protokolleri için henüz bir standart mevcut değildir. Literatürde bu konuda yapılan çok sayıda çalışma yer almaktadır. Önerilen MAC protokollerinin birincil amacı, KAA'nın yaşam süresini de belirleyen enerji tüketimini en aza indirmektir. Bunun yanı sıra, gecikmelere karşı duyarlı olan uygulamalarda mesaj gecikmelerini azaltmak için de çalışmalar yapılmaktadır.

KAA'da önerilen MAC protokolleri merkezi ve çekişme tabanlı olmak üzere ikiye ayrılmaktadır. Merkez tabanlı olarak TDMA kullanılırken, çekişme tabanlı olarak standart DCF (CSMA\CA) protokolleri kullanılmaktadır.

4. BAŞARIM DEĞERLENDİRMESİNDE KULLANILAN KAA MAC PROTOKOLLERİ

4.1. Giriş

KAA'da enerji tüketimini ve gecikmeleri indirgemek için çok sayıda çalışma yapılmış ve çeşitli MAC protokolleri önerilmiştir. Çalışmanın bu bölümünde TDMA tabanlı EDSMAC protokolü ile çekişme tabanlı standart DCF protokolüne enerji duyarlı hali anlatılarak OPNET Modeler yazılımı ile gerçekleştirilen benzetim modelleri sunulmaktadır.

Benzetimi gerçekleştirilen EDSMAC protokolünde TDMA alt yapısı kullanılmıştır. Gecikmelere karşı duyarlı olan uygulamalarda, ek slot tahsisi yardımıyla, gecikmeler indirgenirken, gecikmelerin önemli olmadığı uygulamalarda AD'ler belirli aralılıklarda kapatılarak enerji tüketiminde verimlilik sağlanmaktadır.

Çekişme tabanlı ve enerji duyarlı DCF protokolünde ise CSMA/CA alt yapısı kullanılmıştır. Bu protokol enerji tüketimini indirgemeyi ön planda tutmaktadır. AD'ler belirli zaman aralıklarında uyku moduna alınarak kapatılmaktadır. Bu sayede enerji tüketiminde verimlilik elde edilmektedir.

Bu bölümde sistem, model ve benzetim kavramları, OPNET Modeler programı hakkında temel bilgiler ve tez çalışmasının temelini oluşturan EDSMAC ve enerji duyarlı DCF protokolünün detaylı anlatımı bulunmaktadır.

4.2. Sistem, Model ve Benzetim Kavramları

Bir sınır içerisinde, birbirleriyle etkileşim içinde bulunan ve ortak bir amaca yönelmiş olan elemanlar topluluğudur. Sistem, girdileri çıktılara dönüştüren birbirleriyle ilişkili faaliyetlerden ve elemanlardan oluşmaktadır.

Bir sistemin değişen koşullar altındaki davranışlarını incelemek, kontrol etmek ve geleceği hakkında varsayımlarda bulunmak amacıyla elemanları arasındaki bağlantıları kelimeler veya matematiksel terimlerle belirleyen ifadeler topluluğuna model denir.

Endüstri ve sanayide modellerin kullanılma sebepleri, maliyetlerinin düşüklüğü, tehlikeli olmayışları ve gerçek sistemler üzerinde deney yapmanın bazen imkansızlaşmasıdır. Gerçek sistemlere benzer modeller üzerinde deney yapmak, para ve zaman tasarrufu demektir.

Benzetim, somut anlamda belirli bir nesnenin modeli veya temsilidir. Daha geniş bir ifade ile, gerçek sistemin modelinin tasarımı ve bu model ile sistemin işletilmesi amacına yönelik olarak, sistemin davranışını anlayabilmek veya değişik stratejileri değerlendirebilmek için deneyler yürütülmesi sürecidir [18].

Karmaşık elektronik ve haberleşme sistemlerinin tasarımı, geliştirilmesi, test edilmesi, eğitimi, v.b. artık özel donanım ve yazılımlarla bilgisayar ortamında gerçeklenir olmuştur. Modelleme ve Benzetim; tasarlanacak olan sistemin henüz elde olmadığı, gerçek test ve ölçülerin tehlikeli ve/veya pahalı olduğu ve gerçek test ve denemelerin yapılamadığı durumlarda vazgeçilmez bir araç haline gelmiştir.

Benzetim deneysel bir yöntemdir. Gerçek sistem ile deney yapmak yerine deneyler benzetim modeli üzerinde yapılır. Benzetimin birçok dezavantajları vardır. En önemli dezavantajı benzetim modelinin yaratılmasındaki yoruculuk ve benzetim modelinin programlanmasında kullanılan dilin (örneğin Pascal gibi) zorluğudur. Uygun benzetim yazılımları bulunmaktadır, ancak maliyetlerinin yüksekliği nedeniyle alımı güçleşebilmektedir. Bazı grafik tekniklerine dayanan benzetim yazılımları geliştirilmiştir. Bu yazılımlar sayesinde belirli sistemler için benzetim modellerinin oluşturulması otomatik olarak yapılabilmektedir.

4.3. OPNET Benzetim Yazılımı

OPNET Modeler, kablolu ve kablosuz ağların modellenerek benzetimlerinin yapılmasına olanak sağlayan bir yazılımdır. OPNET Modeler yazılımı Unix veya Windows NT \ 2000 \ XP işletim sistemlerinde çalışmaktadır. OPNET Modeler, sırayla sistemin modellenmesi, veri toplama, benzetimin yapılması ve benzetim sonuçlarının analiz edilme işlemlerini gerçekleştirmektedir [22].

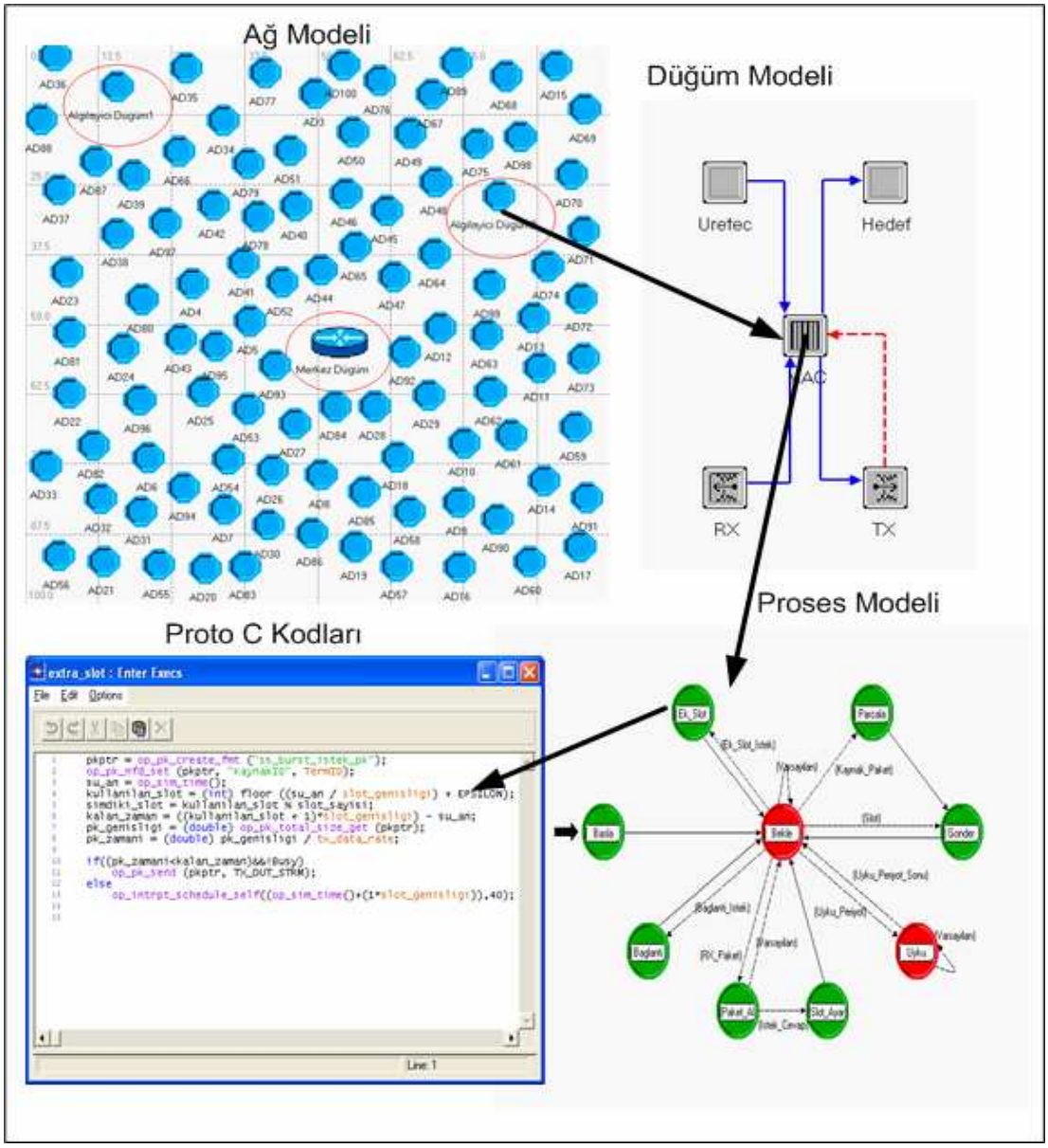
OPNET Modeler yazılımı, ATM, TCP/IP, frame relay, MPLS, IP gibi ağ protokollerinin ve 3Com, Cisco, Bays Network gibi firmaların üretmiş olduğu anahtar ve yönlendirici gibi ürünlerin modellerinin bulunduğu geniş bir kütüphaneye sahiptir. Bunun yanı sıra OPNET'in en büyük avantajı kullanıcıların yeni protokol ve ürünlerin modellerini oluşturabilmesidir.

OPNET Modeler yazılımında bir sistemin modellenmesi bir çok editör yardımıyla olmaktadır ve bu editörler arasında da hiyerarşik bir yapı söz konusudur. OPNET Modeler yazılımında yaygın olarak kullanılan editörler ve görevleri aşağıdaki gibidir [22];

- Proje editörü (Project editor): Ağ modellerinin geliştirildiği, alt ağlar, bağlantı hatları, düğümler (nodes) ve coğrafik içerik tanımlamalarından oluşan editördür. Bu editör temel benzetim ve analiz yeteneklerini içerir.
- Düğüm modeli editörü (Node model editor): Proje editöründe tasarlanan ağ modellerindeki nesnelerin (düğümlerin) geliştirildiği editördür. Düğüm editörünün içerisinde proses modellerinden oluşan modüller ve bu modülleri birbirine bağlayan iletim hatları bulunur. Düğüm modelinde alıcı, verici, kuyruk, işlemci ve üreteç modülleri bulunmaktadır.
- Proses model editörü (Process model editor): Düğüm model editöründe bulunan nesnelerin yapısının, işlevlerinin, parametre ve davranışlarının kontrol edildiği ve çeşitli düzenlemelerin yapıldığı editördür. Bu editör durum geçiş diyagramlarını içermektedir. Ayrıca durum geçiş diyagramları Proto-C kodlarını içermektedir.
- Bağlantı model editörü (Link model editor): Ağ cihazlarının iletişimini sağlayan bağlantı modellerinin oluşturulduğu ve düzenlendiği editördür.

- Paket biçim editörü (Packet format editor): Modellemesi yapılacak sistemde kullanılacak olan veri ve kontrol paketlerinin tanımlandığı ve düzenlendiği editördür.
- Arayüz kontrol bilgisi editörü (Interface control information editor): Prosesler arasında haberleşme kontrol bilgisini tanımlamak için kullanılır.
- Anten model editörü (Antenna pattern editor): Kablosuz modülde bulunan alıcı ve verici için anten örnekleri oluşturmak ve düzenlemek için kullanılmaktadır.
- Modülasyon editörü (Modulation curve editor): Alıcı ve vericiler için modülasyon işlemlerini düzenlemekte kullanılan editördür. Sadece kablosuz modül desteği eklenmiş OPNET yazılımlarında bulunmaktadır.
- PDF editörü (Probability Density Function editor): Olasılık sıklık fonksiyonlarının düzenlendiği editördür. OPNET Modeler yazılımı içerisinde rasgele değerler için çok sayıda analitik dağılım fonksiyonları (Exponential, Poisson, Gamma, Uniform v.b.) bulunmaktadır.

OPNET Modeler yazılımı ile modellenen sistemlerde genellikle, aralarında hiyerarşik bir yapı olan proses, düğüm ve proje editörleri kullanılmaktadır. Şekil 4.1'de OPNET Modeler yazılımı ile modellenmiş bir ağ sisteminin hiyerarşik yapısı görülmektedir. Ağ modelinde bulunan her bir nesne düğüm editörü içinde tasarlanan modüllerden oluşmaktadır. Düğüm editörü içinde bulunan modüllerde ise proses editörü içerisinde ağ elemanlarının davranışlarını düzenleyen geçiş diyagramları ve Proto C kodları bulunmaktadır. Proje editörü modellemenin en üst katmanını oluştururken, proses editörü ise en alt katmanını oluşturmaktadır.



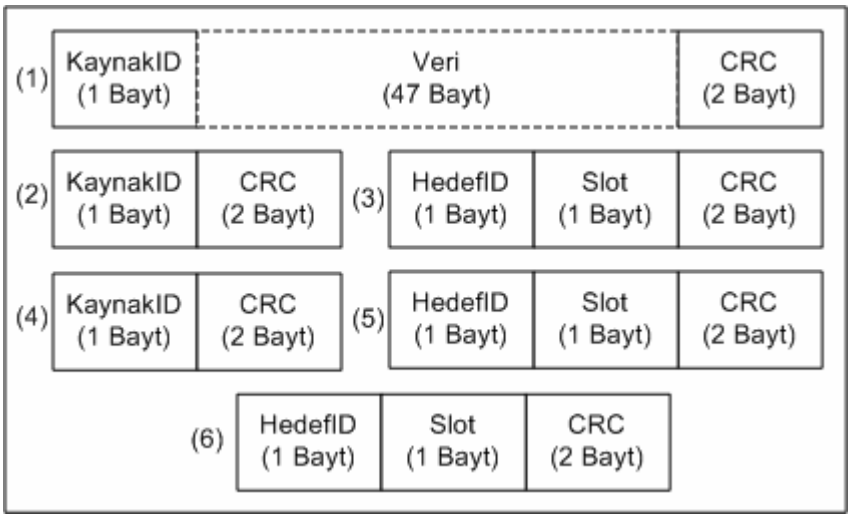
Şekil 4.1: OPNET sistem modellerinin tasarımında kullanılan editörlerin hiyerarşik yapısı

4.4. Enerji ve Gecikme Duyarlı EDSMAC Protokolü

KAA'da gecikmeleri ve enerji tüketimini indirgemek için önerilen MAC protokolüdür [11]. AD'ler MD üzerinden birbiriyle haberleşmektedir. TDMA tabalı bir MAC protokolüdür. İletim zamanı, slotlara bölünmekte ve her AD kendine ait slot zamanında paketini göndermektedir. Gecikme duyarlılığı olan AD'ler için fazladan slotlar tahsis edilmektedir.

Güç tüketimini azaltmak için gecikme duyarlılığı olmayan AD'ler, belirli aralıklarla uyku moduna geçerler. Bu sayede düğümler enerji tasarrufunda bulunmuş olurlar. Gecikmeye karşı duyarlı olan AD'ler, uyku moduna geçen düğümlere ait slotlar zamanında da iletim yaparlar. Bu özellik sayesinde kuyruktaki beklemeler ve dolayısıyla uçtan uca gecikmeler azalırken, diğer taraftan, AD'nin kullanımı ve güç tüketimi artmaktadır. Bununla birlikte, gecikmeye karşı duyarlı olmayan AD'lerin uyku moduna girmesi sonucunda güç tüketimi indirgenmekte fakat uyku periyodunda iletim gerçekleştirilemediği için kuyruk beklemeleri ve uçtan uca gecikmeler artmaktadır. Yapılan çalışmada, söz konusu protokollerin benzetim modelleri gerçekleştirilerek başarım sonuçları değerlendirilmiştir.

Şekil 4.2'de enerji ve gecikme duyarlı MAC protokolünde kullanılan paket formatları görülmektedir. (1) Veri paketi, (2) Bağlantı istek paketi, (3) Bağlantı cevap paketi, (4) Ek slot istek paketi, (5) Ek slot cevap paketi ve (6) Slot iptal paketi olarak kullanılmaktadır.



Şekil 4.2: Enerji ve gecikme duyarlı MAC protokolünde kullanılan paket formatları

4.4.1. Algılayıcı düğüm MAC modeli

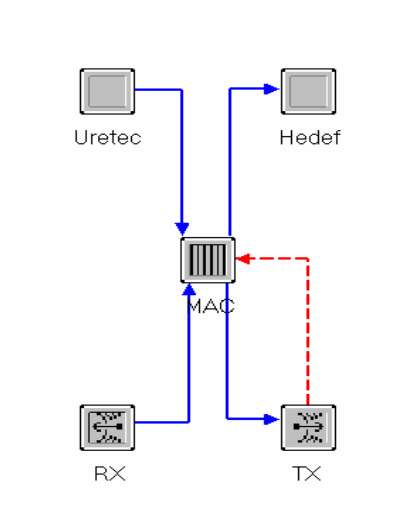
AD için tasarlanan MAC modeli, bağlantının kurulması, gecikme duyarlı trafikler için ek slot isteğinin yapılması, kaynaktan gelen paketlerin parçalara ayrılması, kendi slot zamanında gönderilmesi ve enerjinin daha etkin bir şekilde kullanılması için

gecikmelere karşı duyarlı olmayan düğümlerin uyku moduna geçirilmesi gibi işlevleri yerine getirmektedir.

KAA'da bulunan AD ilk olarak bağlantı isteğinde bulunmaktadır. Bu işlem AD' nin bağlantı istek paketini üretip MD'ye göndermesiyle gerçekleşmektedir. Bu işlem sonunda AD'ye, MD tarafından bağlantı cevap paketi gönderilerek kullanacağı slot bildirilir.

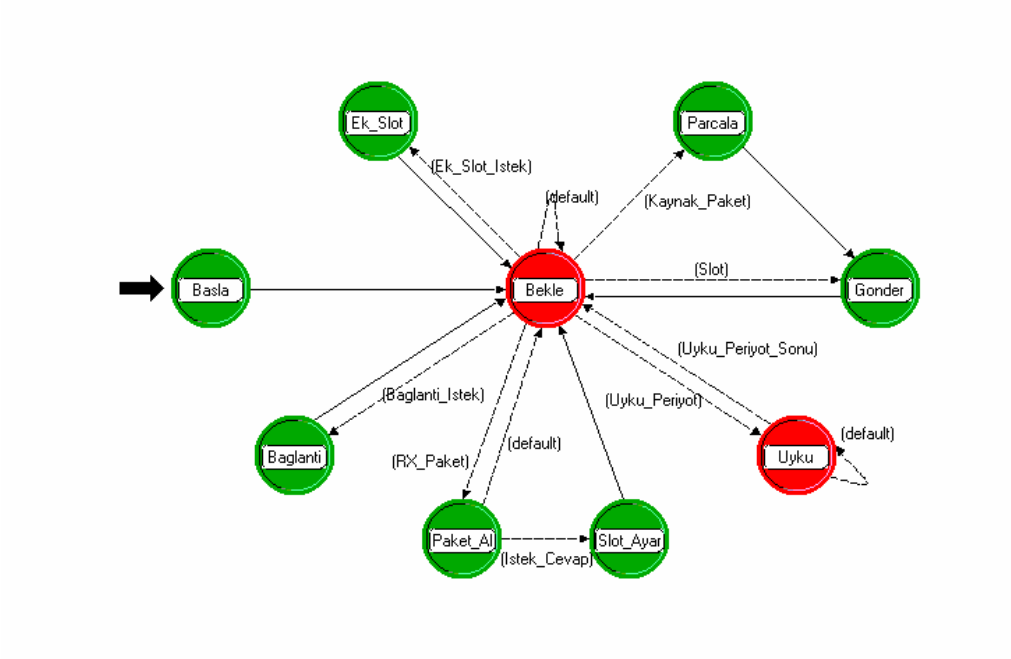
AD üretmiş olduğu paketi göndermek isterse, üretilen paket veri paketi formatında parçalanarak kuyruğa atılmaktadır. Eğer o anda kullanılmakta olan slot AD'ye ait slot ise kuyruktan paket alınıp hedefe gönderilmekte ve bir sonraki zaman dilimine kesme ayarlanmaktadır. Eğer AD'ye ait slot değilse paket gönderilmeden bir sonraki zaman dilimine kesme ayarlanmaktadır. Gecikme duyarlılığı olan AD'lerde kuyrukta bulunan paketler üst eşik değerini geçerse ek slot istek paketi üretilerek MD'den istekte bulunulmaktadır. Bu işlem sonucunda MD ek slot cevap paketi üretmekte ve istekte bulunan AD'ye paketi göndererek tahsis etmiş olduğu ek slotu bildirmektedir. Gecikmelere karşı duyarlı olmayan AD'lerin (radyo iletişim fonksiyonları) belirli zaman aralıklarında kapatılarak uyku moduna geçmeleri sağlanmaktadır. Bu durumda iletim yapılmamakta, fakat üreteçten gelen paketler parçalanarak kuyruğa gönderilmektedir. Uyku modunun bittiğini belirten kesme ile yeniden iletim başlamaktadır.

Şekil 4.3'te AD için tasarlanan düğüm modeli görülmektedir. Düğüm modeli üreteç, hedef, MAC, RX (alıcı) ve TX (verici) elemanlarından oluşmaktadır. Üreteç tarafından üretilen veri paketleri MAC katmanına iletilmektedir. Paketler MAC katmanındaki işlemlerden sonra TX üzerinden MD'ye gönderilir. MD'den gelen paketler RX tarafından alınmakta ve MAC katmanı üzerinden hedefe göndermektedir. AD için tasarlanan proses model MAC katmanı içindedir.

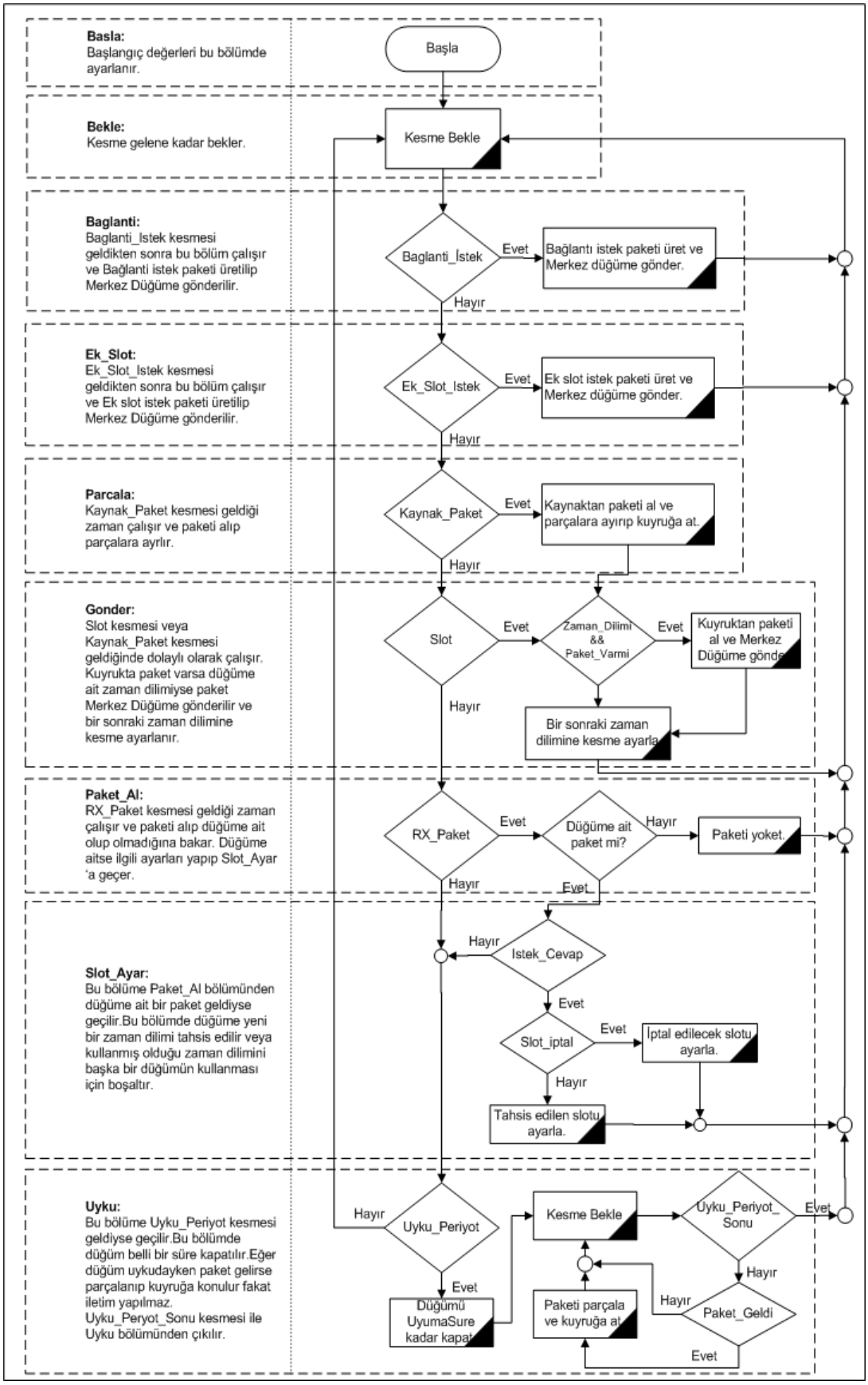


Şekil 4.3: AD'nin düğüm modeli

Şekil 4.4'de OPNET Modeler yazılımında AD için tasarlanan MAC katmanı proses modeli görülmektedir. Şekil 4.5'te bu modelde kullanılan fonksiyonların açıklaması ve akış şeması bulunmaktadır.



Şekil 4.4: AD'nin MAC katmanı proses modeli



Şekil 4.5: AD için tasarlanan MAC katmanı proses modelinin akış şeması

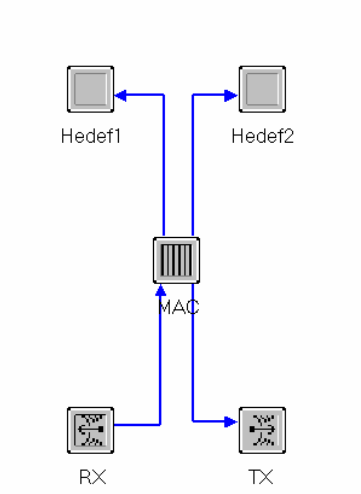
4.4.2. Merkezi düğüm MAC modeli

MD, AD'nin göndermiş olduğu paketleri alıp biçimine bakarak değerlendirmeler yapmaktadır. MD'ye gelen paketler; veri, bağlantı istek veya ek slot istek paketi olup, bunun dışında bir paket geldiyse yok edilmektedir. Veri paketi gelmesi durumunda paketin hedef bilgisine bakılarak istatistiksel bilgileri alınmak üzere ilgili modüle iletilmektedir.

MD'ye gelen paket bağlantı istek paketiyse, slotların hangi terminal tarafından kullanacağını saklayan slot tablosu taranıp bağlantı isteği için ayrılan slotlardan boş bulunan ilk slot tahsis edilmekte ve slot tablosu güncellenmektedir. Bu işlem sonunda AD'ye bağlantı cevap paketi gönderilerek tahsis edilen slot bildirilmektedir.

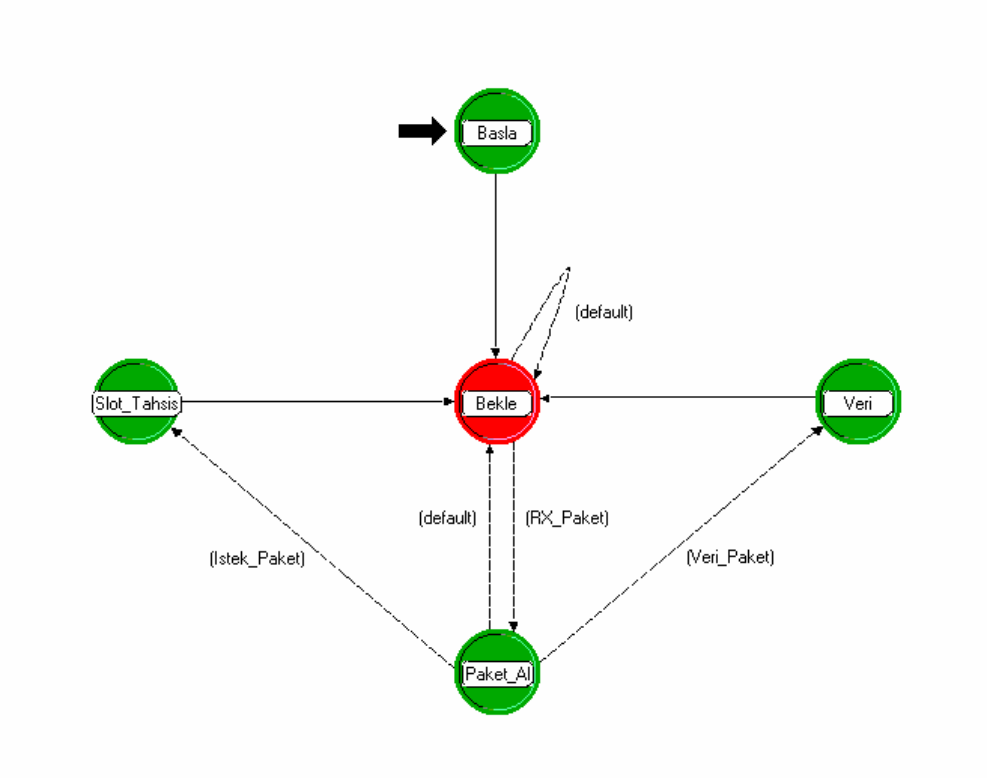
MD'ye gelen paket ek slot istek paketiyse, ek slot tahsisi için ayrılan bölümde boş slot varsa tahsis edilmekte ve AD'ye ek slot cevap paketi ile bildirilmektedir. Eğer ek slot tahsisi için ayrılan bölümde boş slot yoksa, uniform dağılım kullanılarak rasgele bir slot tahsis edilmektedir. Bu işlemden sonra AD'ye ek slot cevap paketi ile kullanacağı ek slot bildirilmektedir. Ayrıca daha önceden slotu kullanan AD'ye ise slot iptal paketi gönderilerek slotun iptal edildiği bildirilmektedir.

Şekil 4.6'da MD için OPNET Modeler yazılımında tasarlanan düğüm modeli görülmektedir. MD için tasarlanan düğüm modeli RX, TX, Hedef1 ve Hedef2 elemanlarından oluşmaktadır. AD tarafından gönderilen paketler MD'nin RX modülü tarafından alınıp, MAC katmanına iletilmektedir. MAC katmanına gelen paket, veri paketi olursa istatistiksel bilgileri alınmak üzere Hedef1 veya Hedef2 modülüne, farklı bir paketse AD düğümlere iletmesi için TX modülüne göndermektedir.

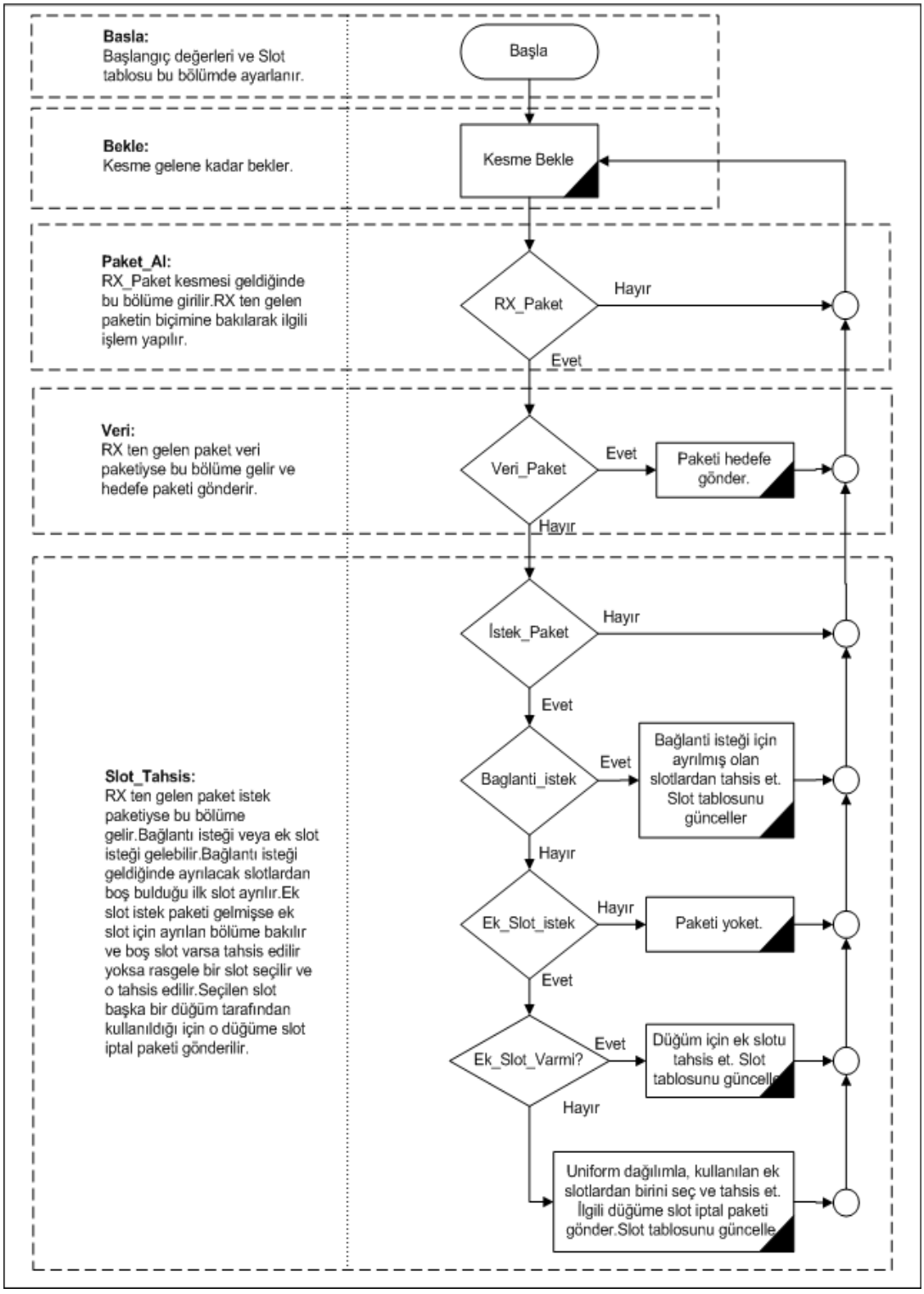


Şekil 4.6: MD'nin düğüm modeli

Şekil 4.7'de OPNET Modeler yazılımında MD için tasarlanan MAC katmanı proses modeli görülmektedir. Şekil 4.8'de ise bu modelde kullanılan fonksiyonların açıklaması ve akış şeması bulunmaktadır.



Şekil 4.7: MD'nin MAC katmanı proses modeli



Şekil 4.8: MD için tasarlanan MAC katmanı proses modelinin akış şeması

4.5. Enerji Duyarlı DCF Protokolü

Tez çalışmasında, çekişme tabanlı yapıya örnek olarak enerji duyarlı DCF protokolü OPNET Modeler yazılımı sayesinde modellenmiştir. AD'ler belirli periyotlarla çalışmakta ve uyku moduna alınmaktadır. AD'lerin uyku moduna geçip geçmeyecekleri baştan ayarlanmaktadır.

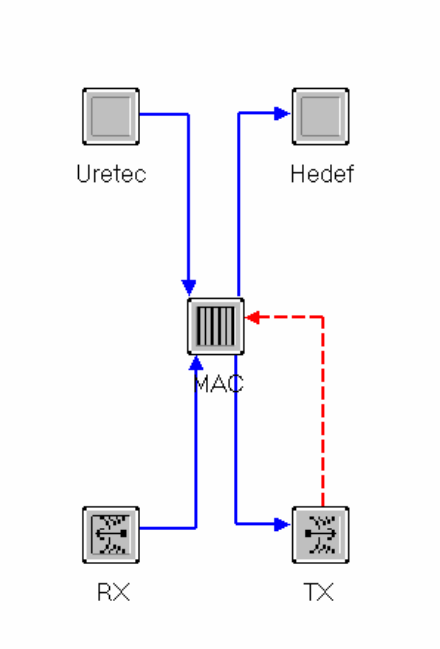
AD'ler iletim ortamını kullanmak için birbiriyle çekişmeye başlarlar. İletim ortamını ele geçiren paketini göndermektedir. İletim ortamının dolu olması durumda, iletim yapmak isteyen AD'ler backoff modülüne girmekte ve ortamı ele geçirene kadar iletimlerini ertelemektedirler. İletim ortamını ele geçiren AD paketini gönderir ve backoff zamanını sıfır ('0') yapmaktadır.

4.5.1. Algılayıcı düğüm MAC modeli

AD'lerin iletim ortamlarını ele geçirmeleri için OPNET Modeler yazılımı sayesinde MAC modeli tasarlanmıştır. AD'ler ilerim ortamını ele geçirdikten sonra veri paketinin kopyası gönderilmektedir. Paketin alınıp alınmadığının anlaşılması için karşı taraftan ACK paketi beklenir. Eğer belli bir süre sonunda ACK paketi gelmezse veri paketinin kopyası tekrardan gönderilmektedir. Eğer ACK paketi, veri paketini gönderen AD'ye gelirse paket kuyruktan alınarak yok edilmektedir.

AD'nin düğüm modeli, üreteç, hedef, RX, TX ve MAC modüllerinden meydana gelmektedir. Üreteç modülü, paketleri üst katmandan alıp MAC modülüne göndermek için kullanılmaktadır. RX modülü, diğer AD'lerin göndermiş olduğu paketleri MAC Modülüne iletmek için kullanılır. TX modülü, MAC modülü üzerinden almış olduğu paketleri diğer AD'lere göndermek için kullanılır. MAC modülü, Üreteç ve RX'ten almış olduğu paketleri, TX veya Hedef modülüne göndermek için kullanılmaktadır. Ayrıca iletim ortamının değişip değişmediğini gösteren TX modülünden MAC modülüne doğru bir bağlantı bulunmaktadır. Bu bağlantı sayesinde ortamın meşgul olup olmadığı ve çeşitli istatistiklerin alınması sağlanmıştır.

Şekil 4.9'da AD için OPNET Modeler yazılımında tasarlanan düğüm modeli görülmektedir. İletim ortamını kullanan bütün AD'ler aynı düğüm modelini kullanarak birbiriyle iletişim kurmaktadırlar.



Şekil 4.9: AD'nin düğüm modeli

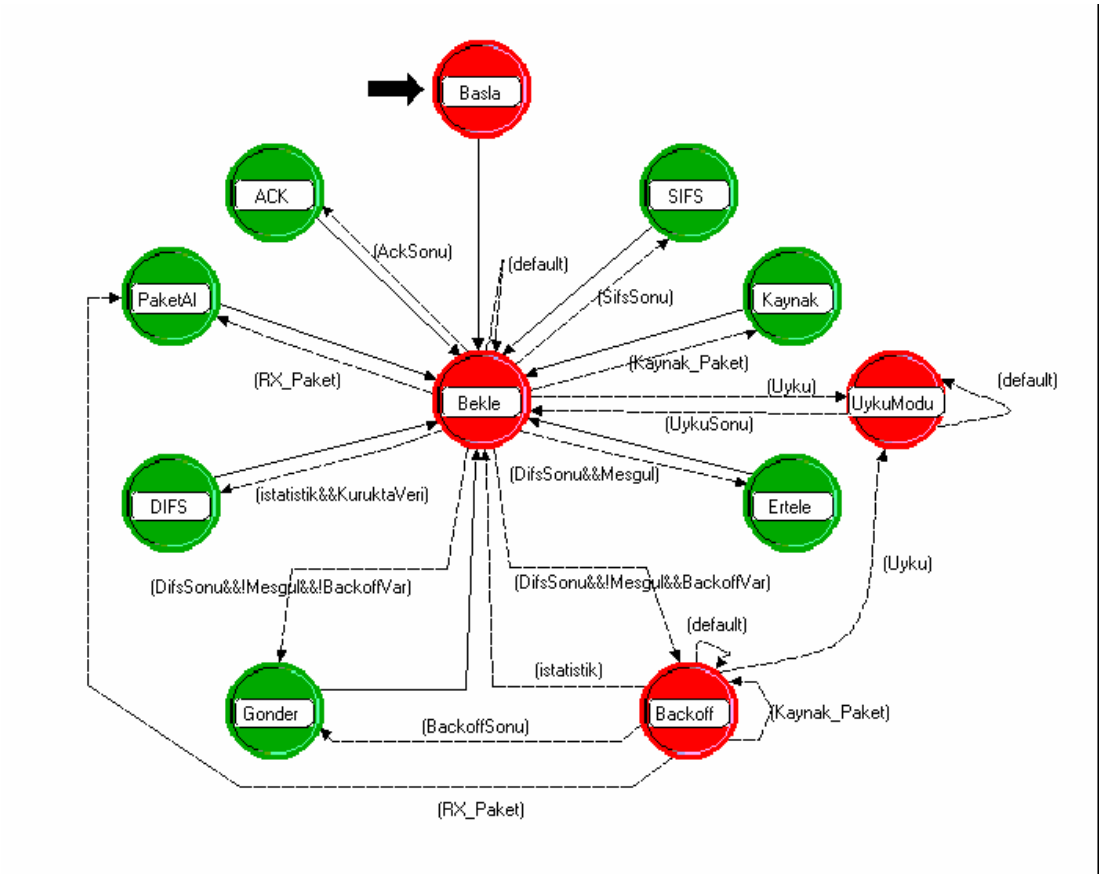
AD'nin MAC katmanı proses modeli ise aşağıda belirtilen modüllerden meydana gelmektedir;

- Basla : Benzetim de kullanılan çeşitli değişkenlere ilk atamaların yapıldığı, istatistiksel değişkenlerin ayarlandığı ve AD'nin uykuya girip girmeyeceğinin belirlendiği modüldür.
- Bekle : Bu modülde, bir olayın gerçekleşmesi beklenmektedir. (Kesme (interrupt) gelmesi beklenir.) Oluşan olaya göre bu modülden ilgili modüle geçiş sağlanmaktadır.
- Kaynak : Bu modüle, üreteç tarafından paket üretildiğinde geçilmektedir. Modül içerisinde, üreteç tarafından gelen veri paketi küçük parçalara ayrılarak kuyruğa atılmaktadır. Daha önceden DIFS kesmesi ayarlanmamışsa,
  - Ortam meşgul değilse DIFS süresinin sonuna DIFS kesmesi ayarlanır
  - Ortam meşgulse BackoffVar özelliği ayarlanmaktadır.

- PaketAl : Bu modüle, RX'ten paket gelince geçilmektedir. Gelen paketin hedefine bakılmakta ve AD'ye ait değilse yok edilmektedir. AD'ye ait olan paketlerin biçimi (formatı);
  - Veri paketi geldiyse, istatistiksel bilgileri alınmak üzere hedefe (sink) gönderir ve SIFS süresi sonuna SifsSonu kesmesi ayarlanmaktadır.
  - ACK paketi geldiyse, veri paketi kuyruktan alınıp yok edilir. ACK paketinin gelmesi, veri paketin diğer AD'ye başarılı bir şekilde ulaştığını göstermektedir.
- Gonder : Bu modül, iletim ortamı ele geçirildiği zaman çalışmaktadır. İletim ortamı AD tarafından ele geçirildiği zaman veri paketinin kopyası gönderilmektedir. Ack süresinin sonuna AckSonu kesmesi ayarlanmaktadır. Eğer bu süre sonunda ACK gelmemişse aynı paket tekrardan gönderilmektedir.
- Backoff : Bu modüle, Difs süresi bitiminde ortam meşgul olmadığı halde backoff gereksinimi varsa girilir. Daha önceden backoff ayarı yapılmadıysa ilk ayarlama yapılmaktadır. Eğer önceden backoff ayarlanmışsa bu bölümde güncellemesi yapılmaktadır. Bu modülde çeşitli olaylar meydana gelmektedir;
  - BackoffSonu kesmesi geldiyse, Gonder modülüne geçilerek iletim yapılır.
  - RX'ten paket geldiyse, PaketAl modülüne geçilerek RX'ten gelen paket alınıp gerekli işlem yapılır.
  - Kaynaktan paket geldiyse, veri paketi parçalanarak kuyruğa atılmaktadır.
  - İletim ortamının değiştiğini gösteren istatistik kesmesi geldiyse Bekle modülüne geçilmektedir.
  - Uyku kesmesi geldiyse, UykuModu modülüne geçilerek AD'nin uyku moduna geçirilmesi sağlanmaktadır.
- UykuModu : UykuModu modülünü, sadece uykuya girecek olan AD'ler kullanmaktadır. Bu modülde, uyuma süresinin sonuna UykuSonu kesmesi ayarlanmaktadır. Bu kesme ile ilgili AD'nin tekrar çalışması sağlanmaktadır. AD'nin tekrar uyuması için uyandığı anda uyuma süresinin sonuna Uyku kesmesi ayarlanmaktadır. UykuModu modülüne girmiş olan AD sadece üst katmandan yani üreteçten veri paketini alarak kuyruğa atmaktadır.
- DIFS : DIFS modülüne, ortamın değiştiği anda kuyrukta veri varsa girilmektedir. İletim ortamı meşgul değilse ve önceden DifsSonu kesmesi ayarlanmamışsa DIFS süresi sonuna bu kesmeyi ayarlar. Eğer daha önceden DifsSonu kesmesi ayarlanmışsa, backoff ihtiyacı olduğunu belirten değişkeni ayarlamaktadır.

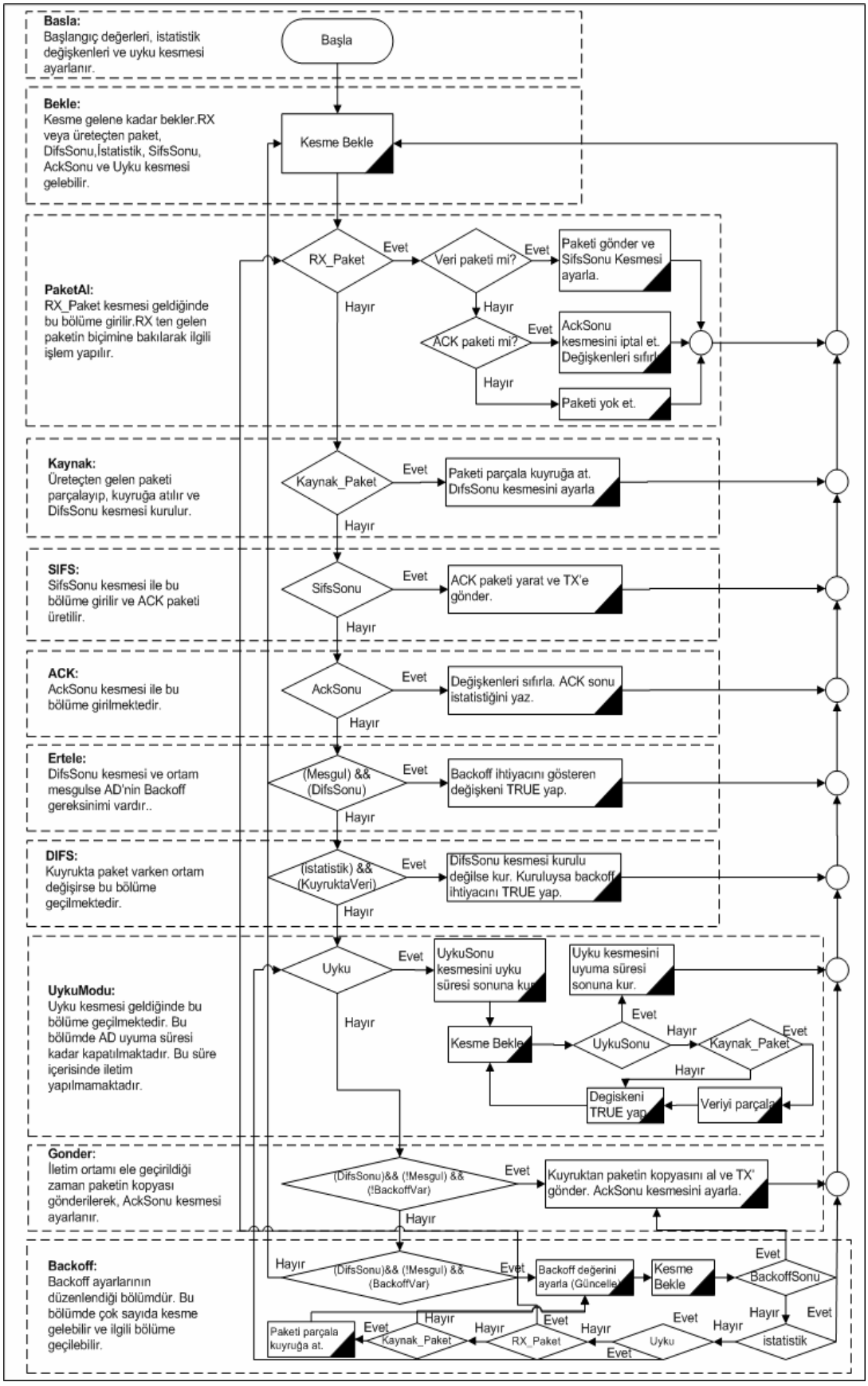
- SIFS (Shortest IFS) : Bu modüle, SifsSonu kesmesi geldiğinde geçilmektedir. SIFS süresinin sonunda ACK paketi yaratılarak, TX'e gönderilmektedir.
- Ertele : DifsSonu kesmesi gelip ortamın meşgul olduğu durumda bu modüle geçilmektedir. Bu durumda backoff ihtiyacı olduğunu belirten değişken ayarlanmaktadır.
- ACK : Bu modüle, AckSonu kesmesi geldiğinde geçilmektedir. Bu modül, paketin başarısız gönderildiğini göstermektedir.

Şekil 4.10'da AD için OPNET Modeler yazılımında oluşturulan düğüm modelinin, MAC modülü içerisinde bulunan proses model görülmektedir.



Şekil 4.10: AD'nin MAC katmanı proses modeli

Şekil 4.11'de AD için tasarlanan MAC katmanının proses modelinde bulunan fonksiyonların işlevlerini gösteren akış şeması bulunmaktadır.



Şekil 4.11: AD için tasarlanan MAC katmanı proses modelinin akış şeması

4.6. Sonuç

Bu bölümde tez çalışmasında kullanılan MAC protokollerinin modelleri detaylı bir şekilde anlatılmıştır. OPNET Modeler yazılımı ile oluşturulan düğüm ve proses modellerin yanı sıra bu proses modellerin çalışmasını anlatan akış şemaları bulunmaktadır.

Merkezi yapıya örnek olarak TDMA protokolünün alt yapısını kullanan EDSMAC protokolü ile çekişme tabanlı yapıya örnek olarak enerji duyarlı DCF protokolünün modellemeleri yapılmıştır. EDSMAC protokolünde AD ve MD için düğüm ve proses modelleri, enerji duyarlı DCF'te ise AD için düğüm ve proses modelleri OPNET Modeler yazılımı sayesinde yapılmıştır.

5. KABLOSUZ ALGILAYICI AĞ BENZETİM SONUÇLARI

5.1. Giriş

Bu bölümde, OPNET Modeler yazılımı kullanılarak, TDMA yöntemine dayalı EDSMAC protokolünü kullanan KAA uygulaması tasarlanmıştır. Ayrıca, örnek uygulamanın başarımı, değişik yük koşulları altında elde edilen benzetim sonuçlarına göre değerlendirilmiştir.

Bununla birlikte, EDSMAC protokolü kullanan KAA uygulaması ile benzer çalışma ve benzetim parametrelerine enerji duyarlı DCF protokolü OPNET Modeler yazılımı ile modellenerek benzetim sonuçları alınmış ve EDSMAC protokolü kullanan KAA uygulaması ile karşılaştırılmıştır.

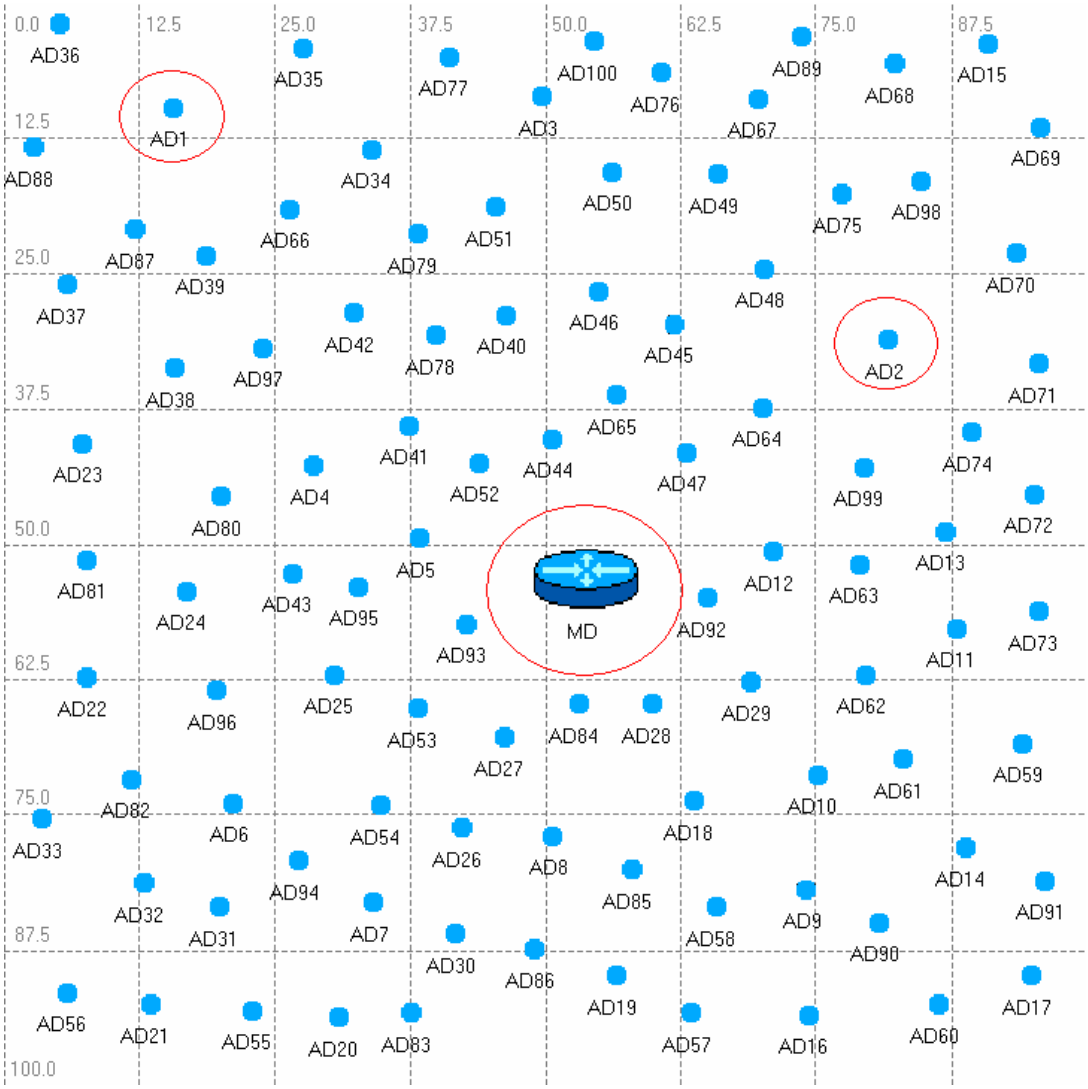
5.2. Benzetim Kabulleri

OPNET Modeler yazılımı ile modellenen ve Şekil 5.1'de gösterilen örnek uygulamada, AD'ler birbirleriyle haberleşmek için Bölüm 4'te detaylı olarak anlatılan EDSMAC protokolünden faydalanırlar. Örnek KAA uygulamasında iki temel bileşen bulunmaktadır. Bunlardan ilki, AD'lerin haberleşmesini kontrol eden MD, diğeri ise olayları algılamak için kullanılan AD'lerdir.

OPNET Modeler yazılımı ile modellemesi yapılan diğer örnek uygulamada ise, AD'ler haberleşmek için Bölüm 4'te detaylı olarak anlatılan enerji duyarlı DCF protokolünden faydalanırlar. AD'ler ortamdaki fiziksel olayları algılamak, verileri işlemek ve merkezi düğüme iletmek için kullanılırlar.

Başarım karşılaştırmasının doğru, geçerli ve anlamlı olması için KAA çalışma koşulları iki örnek uygulamada da benzer seçilmiştir. KAA'yı oluşturan algılama alanı 100*100 metre'dir. EDSMAC protokolü kullanan KAA uygulamasında 100

adet AD ve 1 adet MD bulunurken, diğer KAA uygulamasında ise 100 adet AD kullanılmıştır. EDSMAC protokolü kullanan örnek KAA uygulamasında 100 adet AD'den rasgele 2 tane AD seçilmiştir. Bu AD'lerden AD1 uyku moduna girerek enerji tüketimini indirgerken, AD2 ise gecikmelere karşı duyarlı uygulamalar için kullanılmıştır. Enerji duyarlı DCF protokolü kullanan KAA uygulamasında ise 4 adet AD seçilmiştir. Bunlardan, AD1 – AD2 enerji duyarlı ve AD3 – AD4 ise gecikme duyarlı uygulamalar için kullanılmıştır.



Şekil 5.1: EDSMAC protokolünü kullanan KAA uygulaması

Yapılan deneysel çalışmalar sonunda, EDSMAC protokolünü kullanan KAA uygulamasında, bir slot zamanı 1ms kabul edilmiş ve iletim zamanı 120 slot'a bölünmüştür. Yapılan deneysel çalışmalar sonunda, ilk 100 slot bağlantı isteği

geldiğinde 100 adet AD'ye tahsis edilmiş ve kalan 20 adet slot ise gecikmelere karşı duyarlı olan uygulamalarda kullanılmak üzere ayrılmıştır.

Yine yapılan deneysel çalışmalar sonunda, her slot zamanında 2 adet veri paketi gönderilmesinin gecikmeleri iyileştirdiği gözlenmiştir. KAA uygulamalarında kullanılan benzetim parametreleri Tablo 5.1'de verilmiştir.

Tablo 5.1: Benzetim parametreleri

Parametre	**Değer**
Alan	100 x 100 m
Düğüm Sayısı	100 adet
Güç	AD=10 mW MD=10 mW
Benzetim Çalışma Süresi	60 dk.
Mesaj Boyutu	20 paket x 50* Bayt
Varışlar arası süre (Interarrival Time)	1* – 8* s
Slot Genişliği	1 ms
Modülasyon Tekniği	QPSK
Kuyruk Üst Eşik Değeri	12000 Bit
Veri İletim Hızı(Data rate)	1 Mbit/s
Frekans	RX=3 GHz TX=4 GHz
Slot Sayısı	120 Slot/Frame
Kanal Modeli	Free Space Yayılım Modeli
*Üstel (Exponential) dağılım fonksiyonu kullanılarak üretilmiştir.	

5.3. Benzetim Sonuçları

Yukarıda belirtilen parametreler kullanılarak yapılan benzetim sonuçları bu alt bölümde sunularak başarım değerlendirilmesi yapılmıştır. Elde edilen sonuçların güvenirliliğinin ve hassasiyetinin sağlanması açısından istatistiksel değerler, belirli bir çalışma süresinden sonra alınmıştır.

Örnek uygulamaların başarımlarını değerlendirmek üzere, uçtan-uca gecikme (End To End Delay-EED), kullanım (Utilisation), işlem hacmi (Throughput) ve kuyruk boyutu (Queue Size) gibi değerler alınmıştır.

Uçtan-uca gecikme, paketlerin kaynak tarafından üretildiği andan hedef tarafından yok edildiği ana kadar geçen süreyi ifade eder.

Kullanım (Utilisation), KAA'da bulunan herhangi bir cihazın kullanım oranını göstermektedir. Benzetim sonuçlarının başarım değerlendirilmesi yapılırken, tüketilen enerjiyi bulmak için kullanılmıştır. [12]'deki çalışmada AD'lerde radyo fonksiyonlarının kullanım oranları, bekleme, veri alma ve veri göndermede, deneysel olarak, sırasıyla 1, 1,05 ve 1,4 olarak gözlenmiştir. Enerji tüketimi hesaplanırken bu veriler dikkate alınarak hesaplanmıştır.

Denklem 5.1 ile AD'nin harcamış olduğu toplam enerji bulunmaktadır. Denklemde ifade edilen E_{tx} değeri AD'nin veri gönderirken harcadığı enerji, $E_{(dinleme+alma)}$ değeri AD'nin veri alırken ve beklerken harcadığı toplam enerji ve E_{uyku} değeri ise AD'nin uyku moduna girdiği zaman harcadığı enerjiyi ifade etmektedir. [3,12]'deki çalışmalarda bekleme ve veri alma sırasında harcanan enerjinin aşağı yukarı aynı oldukları ifade etmektedir. Ayrıca denklemde kullanılan E_{uyku} değeri uyku moduna giren düğümlerde $E_{(dinleme+alma)}$ değerinin %1'i olarak hesaplanmaktadır [3].

$$E_{toplam}=E_{tx}+E_{(dinleme+alma)}+E_{uyku} \quad (5.1)$$

Denklem 5.2'de ise AD'nin veri gönderirken tüketmiş olduğu enerji hesaplanmaktadır. Benzetim sonuçlarından elde edilen ortalama kullanım değerinin güç (P) ile çarpılması ile bulunur.

$$E_{tx}=TX_{(Kullanım)}*P \quad (5.2)$$

Denklem 5.3 ve Denklem 5.4'te AD'nin veri alırken ve beklerken tüketmiş olduğu enerji hesaplanmaktadır. Denklem 5.5'te ise AD'ler uyku moduna girerek 50s çalışıp 50s kapatıldıkları için $E_{(dinleme+alma)}$ değerinin yarısı alınmaktadır.

$$\frac{E_{(dinleme+alma)}}{E_{TX}} = \frac{1}{1{,}4} * \frac{(Dinleme + Alma)_{(Kullanıu)}}{TX_{(Kullanıu)}} \quad (5.3)$$

$$E_{(dineleme+alma)} = E_{TX} * \frac{(1 - TX_{(Kullanıu)})}{TX_{(Kullanıu)}} * \frac{1}{1{,}4} \quad (5.4)$$

$$E_{(dineleme+alma)} = (E_{TX} * \frac{(1 - TX_{(Kullanıu)})}{TX_{(Kullanıu)}} * \frac{1}{1{,}4}) * \frac{1}{2} \quad (5.5)$$

E_{uyku} değeri uyku moduna giren AD'ler için söz konusudur. AD'ler uyku modunda çok azda olsa enerji tüketirler. Denklem 5.6'da uyku modunda bulunan AD'nin tüketmiş olduğu enerjiyi hesaplayan formül bulunmaktadır. Uyku moduna girmeyen düğümler için E_{uyku} değeri hesaplanmamaktadır.

$$E_{uyku}=E_{(dileme+alma)}*0{,}01 \quad (5.6)$$

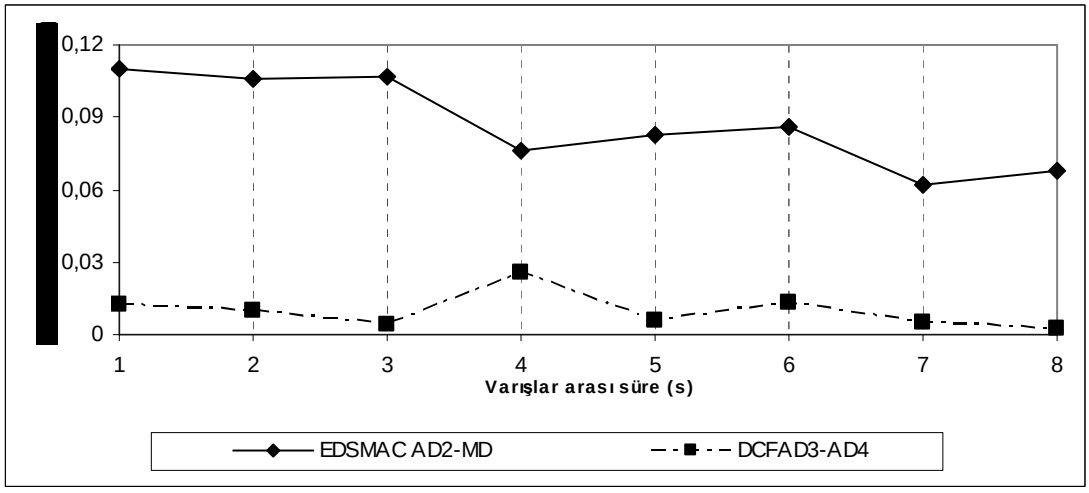
Tablo 5.2'de OPNET Modeler yazılımı ile modellenen EDSMAC ve DCF protokollerinin ortalama uçtan-uca gecikme değerleri gösterilmektedir. EDSMAC protokolünde AD1-MD ve AD2-MD için, enerji duyarlı DCF protokolünde ise benzer koşullarda AD1-AD2 ve AD3-AD4 için ortalama uçtan-uca gecikme değerleri alınmıştır.

Gecikme duyarlı uygulamalarda enerji duyarlı DCF protokolü, EDSMAC protokolüne göre daha iyi sonuçlar vermektedir. Şekil 5.2'de gecikme duyarlı uygulamalar için enerji duyarlı DCF ve EDSMAC protokolleri kullanan örnek uygulamaların ortalama uçtan-uca gecikmeleri görülmektedir. Enerji duyarlı DCF protokolü kullanan gecikme duyarlı örnek uygulamada, en yüksek ortalama uçtan-uca gecikme değeri 0,026s (varışlar arası süre 4s değeri için) olurken, en düşük ortalama uçtan-uca gecikme değeri 0,002s (varışlar arası süre 8s değeri için) olduğu gözlenmiştir. EDSMAC protokolü kullanan gecikme duyarlı örnek uygulamada ise en yüksek ortalama uçtan-uca gecikme değeri 0,11s (varışlar arası süre 1s değeri için) olurken, en düşük ortalama uçtan-uca gecikme değeri 0,062s (varışlar arası süre 7s değeri için) olduğu gözlenmiştir.

Bu bilgilere göre enerji duyarlı DCF protokolünün EDSMAC protokolüne göre gecikme duyarlı uygulamalar için yaklaşık olarak 4 ila 27 kat daha az ortalama uçtan-uca gecikme verdiği gözlenmiştir.

Tablo 5.2: Ortalama uçtan-uca gecikme değerleri

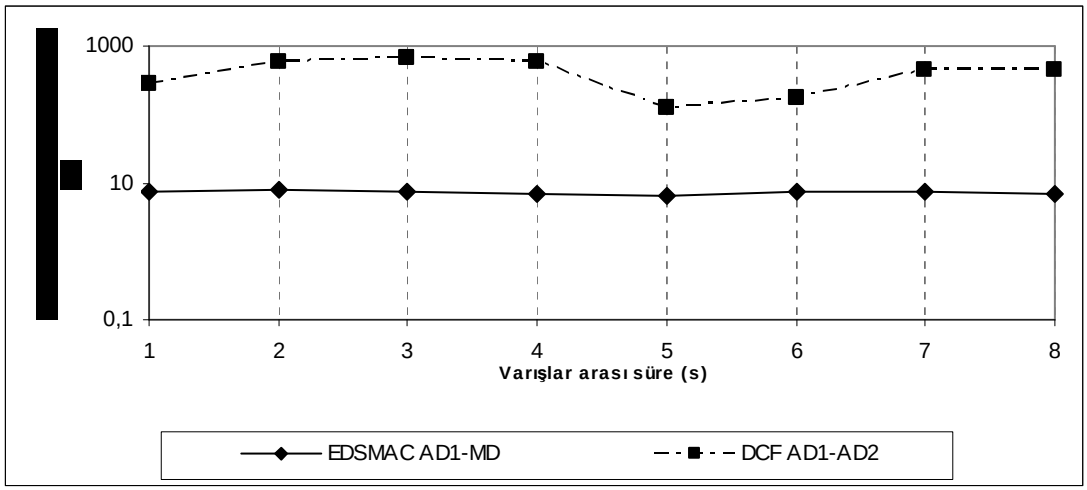
Varışlar arası süre (s)	**EDSMAC AD1-MD (Enerji)**	**EDSMAC AD2-MD (Gecikme)**	**DCF AD1-AD2 (Enerji)**	**DCF AD3-AD4 (Gecikme)**
1	7,194	**0,110**	277,250	0,012
2	**7,738**	0,106	597,484	0,010
3	7,536	0,107	**664,054**	0,005
4	6,810	0,076	573,936	**0,026**
5	**6,338**	0,083	**128,372**	0,006
6	7,551	0,086	178,663	0,013
7	7,176	**0,062**	433,483	0,005
8	6,755	0,068	461,439	**0,002**



Şekil 5.2: Gecikme duyarlı uygulamalarda EDSMAC ve DCF protokollerinin ortalama uçtan-uca gecikmeleri

Şekil 5.3'de enerji duyarlı uygulamalar için EDSMAC ve enerji duyarlı DCF protokollerinin ortalama uçtan-uca gecikmelerini görteren karşılaştırmalı grafik sunulmaktadır. EDSMAC protokolü kullanan enerji duyarlı örnek uygulamada en

yüksek ortalama uçtan-uca gecikme değeri 7,738s (varışlar arası süre 2s değeri için) olurken, en düşük ortalama uçtan-uca gecikme değeri 6,338s (varışlar arası süre 5s değeri için) olduğu gözlenmiştir. Enerji duyarlı DCF protokolü kullanan enerji duyarlı uygulamalarda ise en yüksek ortalama uçtan-uca gecikme değeri 664,054s (varışlar arası süre 3s değeri için) olurken, en düşük ortalama uçtan-uca gecikme değeri 128,372s (varışlar arası süre 5s değeri için) olduğu gözlenmiştir. Bu bilgilere göre, enerji duyarlı uygulamalarda EDSMAC protokolü kullanan örnek uygulama, enerji duyarlı DCF protokolü kullanan örnek uygulamaya göre yaklaşık 20 ila 85 kat daha az ortalama uçtan-uca gecikme verdiği gözlenmiştir.



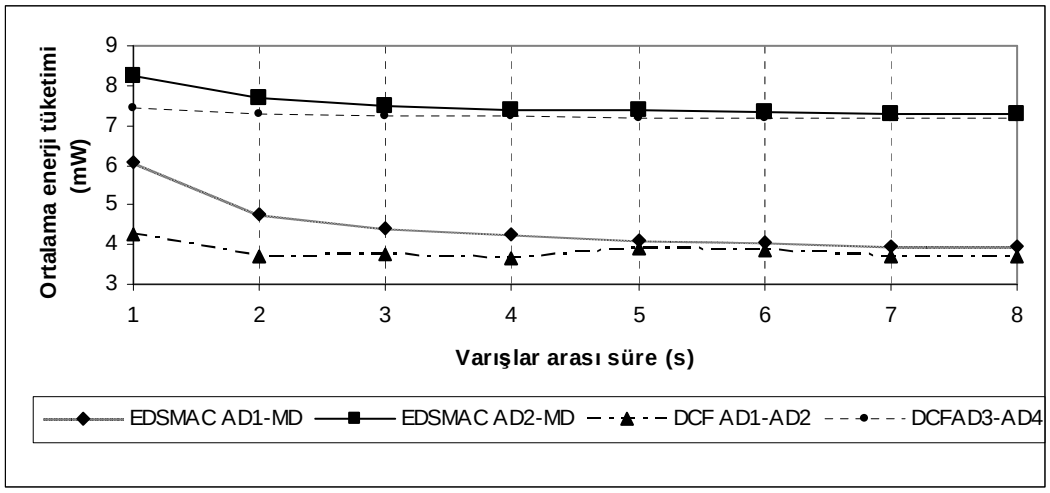
Şekil 5.3: Enerji duyarlı uygulamalarda EDSMAC ve enerji duyarlı DCF protokollerinin ortalama uçtan-uca gecikmeleri

Tablo 5.3 ve Şekil 5.4'te EDSMAC ve enerji duyarlı DCF protokollerinin enerji ve gecikme duyarlı uygulamaları için ortalama güç tüketim değerleri sunulmaktadır. EDSMAC potokolü kullanılan gecikme duyarlı örnek uygulamada en yüksek ortalama enerji tüketimi 8,243mW (1s değeri için) olurken, en düşük ortalama enerji tüketim değeri 7,266mW (8s değeri için) olarak gözlenmiştir. Ayrıca EDSMAC protokolü kullanan enerji duyarlı örnek uygulamalarda en yüksek ortalama enerji tüketimi 6,084mW (1s değeri için) olurken, en düşük ortalama enerji tüketim değeri ise 3,933mW (8s değeri için) olarak gözlenmiştir. Bu bilgilere göre, EDSMAC protokolü kullanan gecikme duyarlı uygulamalar, enerji duyarlı uygulamalara göre yaklaşık olarak 1,3 ila 1,8 kat daha fazla enerji tüketmektedirler.

Enerji duyarlı DCF protokolü kullanan gecikme duyarlı uygulamalarda en yüksek ortalama enerji tüketimi 7,413mW (1s değeri için) olurken, endüşük ortalama enerji tüketimi 7,179mW (8s değeri için) olarak gözlenmiştir. Enerji duyarlı DCF protokolü kullanan enerji duyarlı örnek uygulamada, en yüksek ortalama enerji tüketim değeri 4,243mW (1s değeri için) olurken, endüşük ortalama enerji tüketim değeri 3,680mW (4s değeri için) olarak gözlenmiştir. Bu bilgilere göre, DCF protokolü kullanan gecikme duyarlı uygulamalar, enerji duyarlı uygulamalara göre yaklaşık olarak 1,7 ila 1,9 kat daha fazla enerji tüketmektedir.

Tablo 5.3: Ortalama enerji tüketim değerleri

Varışlar arası süre (s)	**EDSMAC AD1-MD (Enerji)**	**EDSMAC AD2-MD (Gecikme)**	**DCF AD1-AD2 (Enerji)**	**DCF AD3-AD4 (Gecikme)**
1	**6,084**	**8,243**	**4,243**	**7,413**
2	4,084	7,691	3,697	7,286
3	4,395	7,509	3,737	7,235
4	4,240	7,405	**3,680**	7,220
5	4,108	7,369	3,903	7,198
6	4,040	7,324	3,859	7,190
7	3,968	7,298	3,696	7,184
8	**3,933**	**7,266**	3,702	**7,179**



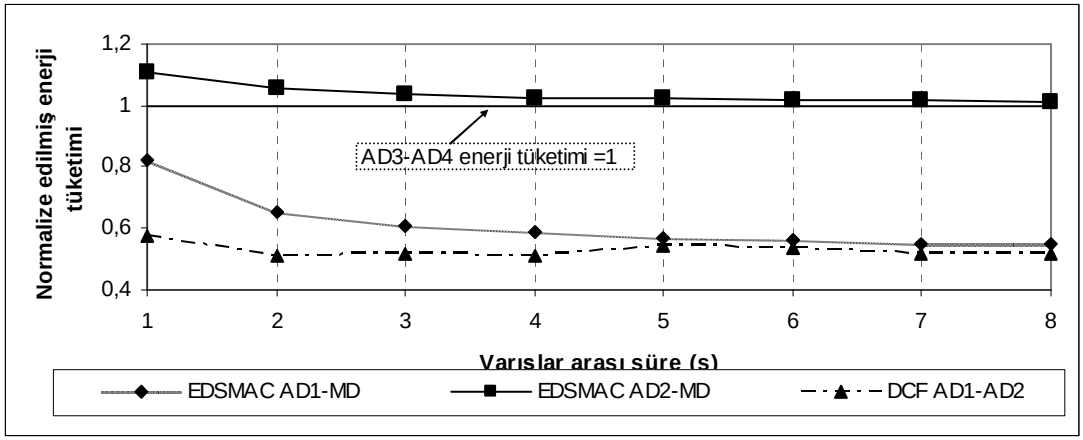
Şekil 5.4: EDSMAC ve DCF protokollerinin güç tüketimi

Enerji duyarlı uygulamalarda DCF protokolü, EDSMAC protokolüne göre yaklaşık olarak 1,1 ila 1,4 kat daha az enerji tükettiği gözlenmiştir. Ayrıca gecikme duyarlı uygulamalarda enerji duyarlı DCF protokolünün, EDSMAC protokolüne göre yaklaşık olarak 1,01 ila 1,1 kat daha az enerji tükettiği gözlenmiştir.

Tablo 5.4 ve Şekil 5.5'te EDSMAC protokolü kullanan AD1-MD, AD2-MD ve DCF protokolü kullanan AD1-AD2'nin güç tüketim değerlerinin enerji duyarlı DCF protokolü kullanan AD3-AD4 eşleniklerine göre normalize edilmiş değerleri ve grafiği sunulmaktadır.

Tablo 5.4: AD1-MD, AD2-MD ve AD1-AD2 güç tüketim değerlerinin DCF AD3-AD4 eşleniklerine göre normalize edilmiş sonuçları

Varış Süresi (s)	**AD1-MD (Uyku Modu)**	**AD2-MD (Gecikme Duyarlı)**	**AD1-AD2 (Uyku Modu)**
1	**0,821**	**1,112**	**0,572**
2	0,653	1,056	**0,507**
3	0,607	1,038	0,516
4	0,587	1,026	0,510
5	0,571	1,024	0,542
6	0,562	1,019	0,537
7	0,552	1,016	0,515
8	**0,548**	**1,012**	0,516



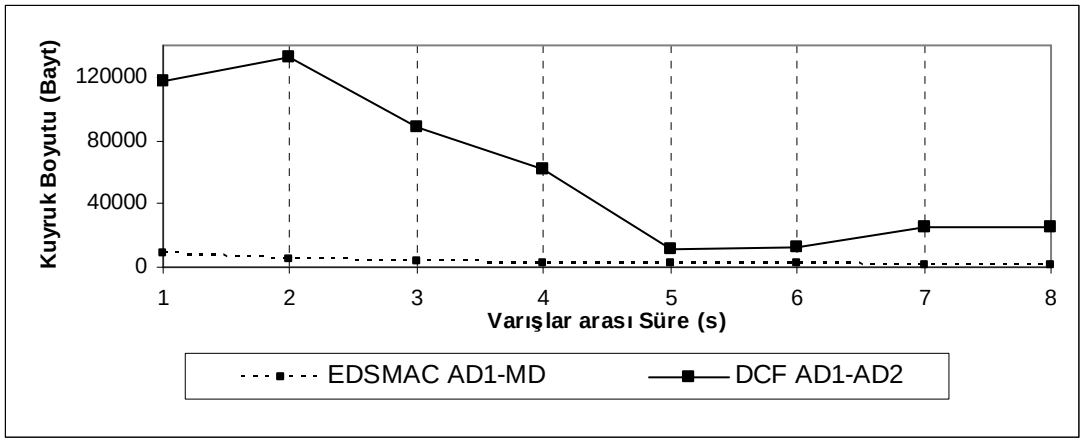
Şekil 5.5: AD1-MD, AD2-MD ve AD1-AD2 güç tüketim değerlerinin DCF AD3-AD4 eşleniklerine göre normalize edilmiş grafiği

Tablo 5.5'de EDSMAC ve DCF protokolleri kullanan örnek uygulamadaki ortalama kuyruk boyutları sunulmaktadır. Enerji duyarlı uygulamalarda DCF protokolünün kuyruk boyutunun çok yüksek olduğu gözlenmiştir. EDSMAC protokolü enerji duyarlı uygulamalarda enerji duyarlı DCF protokolüne göre yaklaşık olarak 7 ila 15 kat daha düşük kuyruk gecikmesi sağladığı gözlenmiştir.

Tablo 5.5: Ortalama kuyruk boyutu değerleri

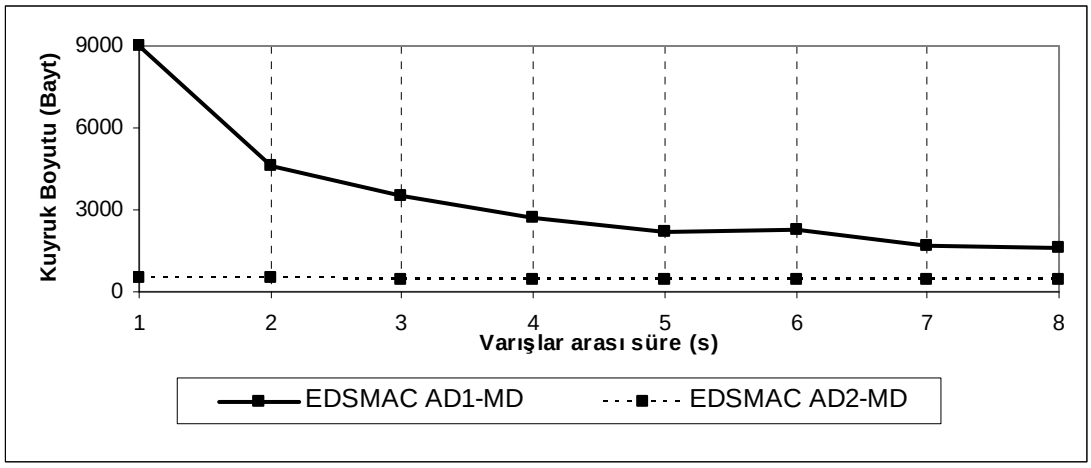
Varışlar arası süre (s)	**EDSMAC (Bayt) (AD1-MD)**	**EDSMAC (Bayt) (AD2-MD)**	**DCF (Bayt) (AD1-AD2)**	**DCF (Bayt) (AD3-AD4)**
1	**8963**	**536**	117637	195
2	4619	484	**132255**	196
3	3477	448	87716	187
4	2691	442	61796	201
5	2207	426	**11368**	**174**
6	2250	467	12857	197
7	1686	410	24865	207
8	**1641**	**405**	25139	**214**

Şekil 5.6'da enerji duyarlı uygulamalarda EDSMAC ve DCF protokollerinin karşılaştırmalı grafiği sunulmaktadır.



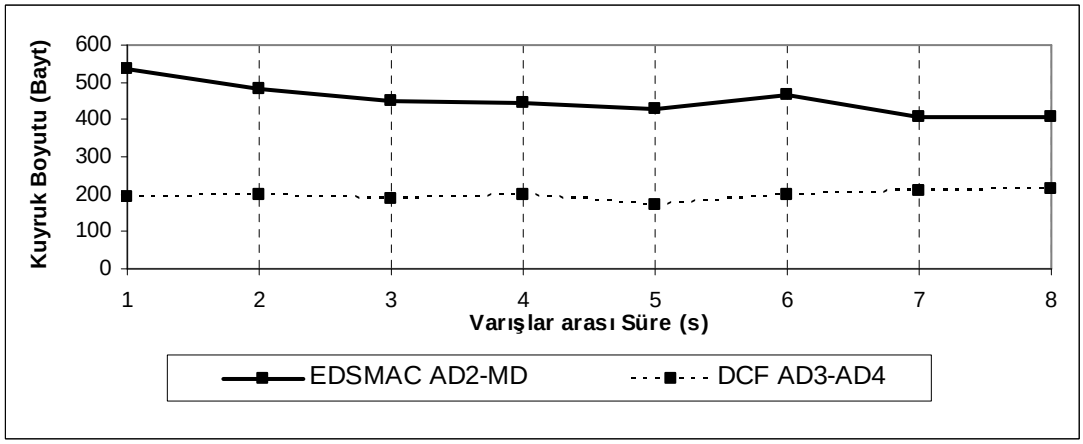
Şekil 5.6: EDSMAC AD1-MD ve DCF AD1-AD2 kuyruk boyutları

Şekil 5.7'de EDSMAC protokolünde gecikme duyarlı uygulamalardaki kuyruk boyutu ile, enerji duyarlı uygulamalardaki kuyruk boyutlarını gösteren grafik bulunmaktadır. Gecikme duyarlı uygulamalarda kuyruk boyutunun, enerji duyarlı uygulamalara göre yaklaşık 4 ila 17 kat daha düşük olduğu gözlenmiştir.



Şekil 5.7: EDSMAC AD1-MD ve AD2-MD kuyruk boyutları

Şekil 5.8'de gecikme duyarlı uygulamalarda EDSMAC ve enerji duyarlı DCF protokollerinin karşılaştırmalı grafiği sunulmaktadır. EDSMAC protokolü gecikme duyarlı uygulamalarda enerji duyarlı DCF protokolüne göre ortalama kuyruk boyutu bakımından yaklaşık olarak 2 ila 2,5 kat daha fazla olduğu gözlenmiştir.

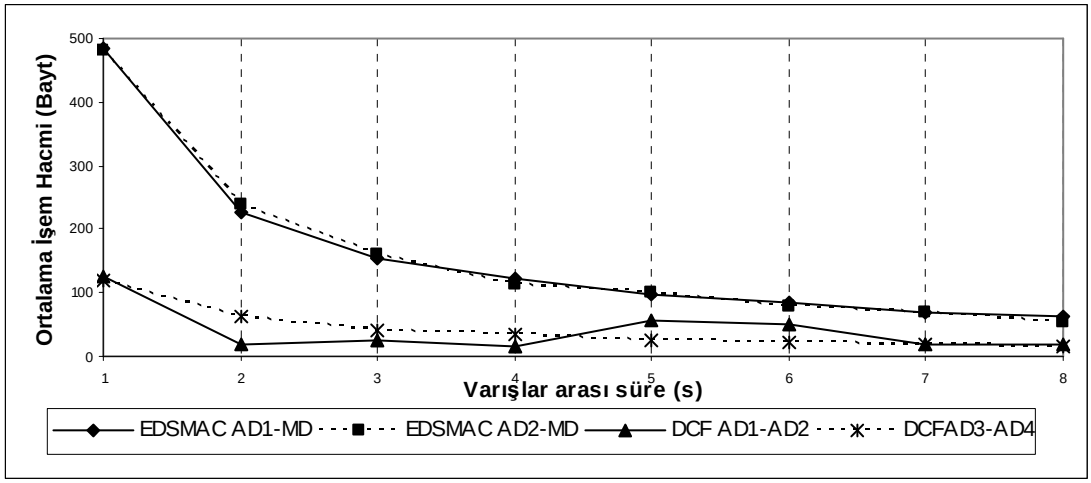


Şekil 5.8: EDSMAC AD2-MD ve DCFAD3-AD4 kuyruk boyutları

Tablo 5.6 ve Şekil 5.9'da EDSMAC ve enerji duyarlı DCF protokollerinin ortalama işlem hacimlerini gösteren tablo ve grafik sunulmaktadır. EDSMAC protokolünün, DCF protokolüne göre enerji ve gecikme duyarlı uygulamalarda yaklaşık 4 kat daha fazla olduğu gözlenmiştir.

Tablo 5.6: Ortalama işlem hacmi değerleri

Varışlar arası süre (s)	**EDSMAC (Bayt) (AD1-MD)**	**EDSMAC (Bayt) (AD2-MD)**	**DCF (Bayt) (AD1-AD2)**	**DCF (Bayt) (AD3-AD4)**
1	**484,25**	**481,45**	**124,378**	**118,385**
2	225,47	239,97	17,624	62,792
3	153,99	160,11	25,299	40,383
4	123,78	114,53	**14,296**	33,588
5	98,00	99,11	57,863	24,244
6	84,64	79,35	49,288	20,443
7	70,57	67,96	17,424	17,921
8	**63,69**	**53,68**	18,498	**15,697**



Şekil 5.9: EDSMAC ve DCF protokollerinin ortalama işlem hacmini gösteren grafik

5.4. Sonuç

Bu bölümde, OPNET Modeler yazılımı ile modellemesi yapılan EDSMAC ve DCF protokollerinin kullanıldığı örnek birer uygulama yapılmış ve benzetim sonuçları alınmıştır.

Enerji duyarlı DCF protokolü gecikme duyarlı uygulamalarda, EDSMAC protokolüne göre yaklaşık olarak 4 ila 27 kat daha düşük ortalama uçtan-uca gecikme vermektedir. Enerji duyarlı uygulamalarda ise EDSMAC protokolü, enerji duyarlı DCF protokolüne göre yaklaşık olarak 20 ila 85 kat daha düşük ortalama uçtan-uca gecikme vermektedir. Enerji duyarlı uygulamalarda gecikmelerin yüksek olması AD'lerin belirli aralıklarla uyku moduna girmesinden kaynaklanmaktadır.

Enerji duyarlı uygulamalarda enerji duyarlı DCF protokolü, EDSMAC protokolüne göre yaklaşık olarak 1,1 ila 1,4 kat daha düşük enerji tükettiği gözlenmiştir. Gecikme duyarlı uygulamalarda ise, iki protokolünde yaklaşık olarak aynı sonuçları verdiği gözlenmiştir. AD'lerin belirli aralıklarla uyku moduna alınarak radyo fonksiyonlarının kapatılmasından dolayı modellemesi yapılan her iki protokoldede enerji duyarlı uygulamalar gecikme duyarlı uygulamalara göre daha iyi sonuçlar verdiği gözlenmiştir.

Enerji duyarlı DCF protokolü, enerji duyarlı uygulamalarda kuyruktaki birikmeler konusunda EDSMAC protokolüne göre yaklaşık olarak 7 ila 15 kat daha yüksek çıktığı gözlenmiştir.

EDSMAC protokolünün, DCF protokolüne göre Enerji ve Gecikme duyarlı uygulamalarda yaklaşık 4 kat daha fazla olduğu gözlenmiştir.

6. SONUÇLAR VE ÖNERİLER

İletişim teknolojisindeki hızlı gelişmelerle beraber, KAA'nın kullanım alanları da hızla artmaktadır. KAA özelikle sağlık, askeri, ticari ve endüstriyel alanlarda yaygın olarak kullanılmaya başlanmıştır. KAA'nın yaygın olarak kullanılmasıyla beraber AD'lerin enerjiyi verimli kullanmaları büyük önem kazanmıştır. AD'lerin enerjiyi verimli kullanmaları KAA'nın yaşam süresini etkilemektedir. Endüstriyel ve akademik alanda KAA için yapılan çalışmalar enerji verimliliğini arttırmak üzerine yoğunlaşmış durumdadır.

Tez çalışmasında, KAA ile ilgili temel bilgiler, önerilen MAC protokollerinin alt yapısını oluşturan TDMA ve CSMA/CA hakkında detaylı bilgiler verilerek OPNET Modeler yazılımıyla merkezi ve çekişme tabanlı MAC protokolleri için birer örnek modellenmiş, ardından bu modellerin benzetimleri yapılarak başarım sonuçları değerlendirilmiştir.

KAA'da kullanılan MAC protokollerinin standardı henüz tanımlanmadığından bu alanda çok sayıda çalışma yapılmaktadır. Önerilen MAC protokolleri merkezi ve çekişme tabanlı olmak üzere ikiye ayrılır. Merkezi yapı olarak TDMA tekniğinin özellikleri kullanılırken, çekişme tabanlı olarak CSMA\CA (DCF) tekniğinin özellikleri kullanılmaktadır.

Tez çalışması çerçevesinde, merkezi tabanlı yapıya örnek olarak TDMA alt yapısını kullanan EDSMAC protokolü modellenmiştir. Bu MAC protokolünde iletim zamanı yapılan deneysel çalışmalar sonunda 120 slot'a ayrılmıştır. Yapılan deneysel çalışmalar sonunda ilk 100 slot bağlantı isteğinde bulunup ortama katılmak isteyen düğümler için ayrılmış, 20 slot ise gecikmelere karşı duyarlı trafiklerin kullanması için ayrılmıştır. Kuyrukta bulunan paketler üst eşik değerini geçtiklerinde algılayıcı düğüm, merkezi düğümden ek slot isteğinde bulunmakta ve 20 slotluk kısımdan ek slot tahsis edilmektedir. Ayrıca enerji tüketimini indirgemek için bazı düğümler uyku

moduna alınarak radyo iletim fonksiyonları kapatılmaktadır. Bu iki özelliği sayesinde mesaj gecikmeleri düşürülerek enerji kullanımında verimlilik sağlanmaktadır.

Yapılan çalışmada, merkezi ve çekişme tabanlı protokol örneklerinin karşılaştırmalı başarımı analizi yapılmıştır. EDSMAC ve enerji duyarlı DCF protokollerinde enerji verimliliğinin artması için, enerji duyarlı olan uygulamalarda AD'ler periyodik olarak uyku moduna alınmaktadırlar. EDSMAC protokolü enerji duyarlı uygulamalarda enerji duyarlı DCF protokolüne göre yaklaşık olarak 20 ila 85 kat daha düşük ortalama uçtan-uca gecikme verirken, gecikme duyarlı uygulamalarda enerji duyarlı DCF protokolü EDSMAC protokolüne göre yaklaşık olarak 4 ila 27 kat daha düşük ortalama uçtan-uca gecikme verdiği gözlenmiştir.

EDSMAC protokolünde enerji duyarlı uygulamaların, gecikme duyarlı uygulamalara göre yaklaşık olarak 1,3 ila 1,8 kat daha az enerji tükettiği gözlenmiştir. Enerji duyarlı DCF protokolü kullanan gecikme duyarlı uygulamalar uyku moduna girmediğinden, enerji duyarlı uygulamalara göre yaklaşık olarak 1,7 ila 1,9 kat daha fazla enerji tüketmektedir. Ayrıca enerji duyarlı uygulamalarda enerji duyarlı DCF protokolü, EDSMAC protokolüne göre yaklaşık olarak 1,1 ila 1,4 kat daha az enerji tükettiği gözlenmiştir.

Enerji duyarlı DCF protokolü enerji duyarlı uygulamalarda kuyruktaki birikmeler konusunda EDSMAC protokolüne göre yaklaşık olarak 7 ila 15 kat daha yüksek çıktığı gözlenmiştir. EDSMAC protokolünün, enerji duyarlı DCF protokolüne göre enerji ve gecikme duyarlı uygulamalarda yaklaşık 4 kat daha fazla olduğu gözlenmiştir.

Tez çalışmasının devamı olarak, EDSMAC protokolünü kullanan düğümlerden uyku modundakilerin slotlarının gecikme duyarlı düğümler tarafından kullanılarak gecikmelerin azaltılması sağlanabilir. Bu çalışmada, uyku modunda olan düğümlerin slotlarına erişmek için DCF yöntemi kullanılabilir. Bu sayede gecikmelere karşı duyarlı olan trafiklerde daha fazla slot kullanılacağından mesaj gecikmesi azaltılabilir.

Çalışmaya konu olan EDSMAC protokolü ve kullanıldığı KAA sisteminin OPNET Modeler benzetim yazılımı ile geliştirilen modeli üzerinde çalışmalar yapılmıştır. DSP yada FPGA gibi yüksek işlem hızına sahip teknolojiler kullanılarak EDSMAC protokolünün ve KAA ağının fiziksel gerçeklemesi çalışmanın devamı niteliğindeki bir başka araştırma–geliştirme projesine konu olabilir.

Kablosuz ortamın en büyük sorunlarından bir diğeri de güvenliktir. KAA sistemlerinde radyo ortamından aktarılan algılanmış verileri içeren paketlerin güvenliğini sağlayacak algoritmalar ve modüller geliştirilebilir. Yine ileriki çalışma olarak KAA yönlendirme algoritmalarının mevcut sistemlere entegre edilmesi düşünülebilir.

KAYNAKLAR

[1] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E., "A survey on sensor networks", ***IEEE Commun.Mag.,*** Vol 40 (8), 102-114, (2002)

[2] Estrin, D., Goyindan, R., Heidemann, J., Kumar, S., "Next CenturyChallenges: Scalable Coordination in Sensor Networks", ***ACM/IEEE International Conference on Mobile Computing and networking***., 263-270, (1999)

[3] Ye, W., Heidemann, J., Estrin, D., "An energy-efficient MAC protocol forwireless sensor Networks", ***Twenty-First Annual Joint Conference of theIEEE Computer and Communications Societies (INFOCOM***), Vol. 3, 1567–1576, (2002)

[4] El-Hoiydi, A., "Spatial TDMA and CSMA with Preamble Sampling for Low Power Ad Hoc Wireless Sensor Networks," ***Proc. ISCC 2002***, 685–692, July 2002

[5] Enz, C.C., El-Hoiydi, A., Decotignie, J-D., Peiris, V., "WiseNET: an ultralowpower wireless network solution", ***IEEE Computer*** ,Vol 37, Issue 8, 244-251, (2004)

[6] Van Dam,T., Langendoen,K., "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks", ***In The First ACM Conference on Embedded Networked Sensor Systems***, Los Angeles,CA,USA, 171–180, November, 2003

[7] Rajendran, V., Obraczka, K., Garcia-Luna-Aceves, J.J., "Energy-Efficient, Collision-Free Medium Access Controlfor Wireless Sensor Networks," ***Proc. ACM SenSys '03***,Los Angeles, CA, 181–192, Nov. 2003

[8] Lin, P., Qiao, C., Wang, X., "Medium Access Control with a Dynamic Duty Cycle for Sensor Networks," ***IEEE WCNC***, Vol. 3, 1534–1539, Mar. 2004

[9] Ye, W., Heidemann, J., Estrin, D., "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," ***IEEE/ACM Trans. Net***., Vol 12 (3), 493–506, (2004)

[10] Nar, P.C., Cayirci, E., "PCSMAC: a power controlled sensor — MAC protocol for wireless sensor Networks", **Second European Workshop on Wireless Sensor Networks**, 81–92., (2005)

[11] Ceken, C., "An energy efficient and delay sensitive centralized MAC protocol for wireless sensor Networks",***ELSEVIER ScienceDirect Computer Standarts & İnterfaces***, (2007)

[12] Stemm, M., Katz, R.H., "Measuring and reducing energy consumption of network interfaces in hand-held devices", ***IEICE Transactions on Communications,*** E80-B (8), 1125–1131, (1997)

[13] Demirkol, I., Ersoy, C., Alagöz, F., "MAC protocols for wireless sensor networks: a survey", ***IEEE Communications Magazine*** ,Vol 44 (4), 115-121, (2006)

[14] Brownfield, M.I., Mehrjoo, K., Fayez, A.S., Davis, N.J., "Wireless Sensor Network Energy-Adaptive MAC Protocol", ***IEEE Communications***., 778-782, (2006)

[15] Polastre, J., Hill, J., Culler, D., "Versatile low power media access for wireless sensor networks." ***In Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys)***, Baltimore,MD, November 2004

[16] Bharghavan, V., et al., "MACAW : a media Access protocol for Wireless LANs", ***ACM SIGCOMM***, Vol. 24.4, 212–225, (1994)

[17] ANSI/IEEE Std 802.11, "Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications", ***IEEE Standards 802.11***, 70–90, (1999)

[18]. Çeken, C., "Kablosuz ATM Kullanarak Servis Kalitesi Desteği Sağlanmış GerçekZamanlı Veri Transferi", Doktora Tezi, ***Kocaeli Üniversitesi Fen Bilimleri Enstitüsü,*** (2004).

[19]. Bayılmış, C., "IEEE 802.11B Klan Kullanarak Can Segmentleri Genişleten Arabağlaşım Birimi Tasarımı", Doktora Tezi, ***Kocaeli Üniversitesi Fen Bilimleri Enstitüsü,*** (2006).

[20]. Kumar, S., Raghavan, V. S., Deng, J., "Medium Access Control protocols for ad hoc wireless networks: A survey", ***ELSEVIER Sience Direct Ad Hoc Networks***, 4, 326-358, (2006)

[21]. International Engineering Consortium, "Time Division Multiple Access (TDMA)", http://www.iec.org/online/tutorials/tdma/index.html, (**Ziyaret tarihi : 18 Aralık 2007**)

[22]. OPNET, "OPNET Modeler 11.5 Documentation", ***OPNET Technologies,*** Release11.5, (2006).

[23] Khemapech, I., Duncan, I., Miller, A., "A survey of Wireless Sensor Networks Technology", ***PGNET***, June 2005

[24] Chong, C-Y., Kumar, S.P., "Sensor Networks : Evolution, opportunities, and challenges", ***Proc IEEE***, Vol 91, No 8, 1247-1256, (2003)

[25] Nicopolitidis, P., Obaidat, M.S., Papadimitriu, G.I., Pomportsis, A.S., "Wireless Networks", ***John Wiley & Sons. Ltd***, 54-67, (2003)

[26] ICT Center, "Wireless Sensor Networks", http://www.ict.csiro.au/page.php?cid=87**,** (**Ziyaret tarihi : 24.12.2007**)

[27] Martincic, F., Schwiebert, L., "Handbook of Sensor Networks : Algorithms and Architectures", ***John Wiley & Sons. Ltd***, 1-36, (2005)

[28] Santi, P., "Topology Control in Wireless Ad Hoc and Sensor Networks", ***John Wiley & Sons. Ltd***, 5-10, (2005)

[29] Haenselmann, T., "SensorNetworks", GNU Free Documentation 5th April 2006, http://www.informatik.uni-mannheim.de/~haensel/sn_book/, (**Ziyaret tarihi: 24.12.2007**)

[30] Callaway, E.H., "Wireless Sensor Networks : Architectures and Protocols", ***CRC Pres***, (2004)

[31] Buettner, M., Yee, G., Anderson, E., Han, R., "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks," ***Proceeding of the 4th ACM International Conference on Embedded Sensor Systems (SenSys)***, 307-320, (2006)

[32] Bernardo, L., Oliveira, R., Pereira, M., Macedo, M., Pinto, P., "A Wireless Sensor MAC Protocol for Bursty Data Traffic", ***Proceedings of the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'07)***, Athens, Greece, September 2007

EK - A : EDSMAC PROTOKOLÜ AD'nin PROSES MODELİNE AİT PROGRAM KODLARI

Bu ekte, EDSMAC protokolünün AD'ler için kullanılan proses modeline ait C program kodları verilmektedir.

"Basla" Durum Makinesi

```
uykusay=0;
uykusayist=op_stat_reg("uykusay",OPC_STAT_INDEX_NONE,OPC_STAT_LOCAL);
slot_genisligi=0.001;
my_id = op_id_self();my_node_id = op_topo_parent (my_id);
op_ima_obj_attr_get (my_node_id, "TerminalID", &TermID);
op_ima_obj_attr_get (my_node_id, "HedefID", &HedefID);
op_ima_obj_attr_get (my_node_id, "UykuModu", &UykuModu);
op_ima_obj_attr_get (my_node_id, "Uretec.Start Time", &baslama);
op_ima_obj_attr_get (my_node_id, "Uretec.Stop Time", &bitis);
/* Transmitter veri hizini al */
tx_id = op_topo_assoc (my_id, OPC_TOPO_ASSOC_OUT, OPC_OBJTYPE_RATX, 0);
comp_id = op_topo_child (tx_id, OPC_OBJTYPE_COMP, 0);
tx_ch_id = op_topo_child (comp_id, OPC_OBJTYPE_RATXCH, 0);
op_ima_obj_attr_get (tx_ch_id, "data rate", &tx_data_rate);
for(i=0;i<dboy;i++)
        benim_slot[i]=-1;
paketGeldi=OPC_FALSE;
replay=OPC_FALSE;
eleman=0;
ciftslot=OPC_FALSE;
slotiptal=OPC_FALSE;
op_intrpt_schedule_self(baslama,10);//baslama zamanina baglanti istegi
if (UykuModu==0) //uyku modu icin
        op_intrpt_schedule_self (op_sim_time ()+baslama+UyumaSure,50);
```

"Baglanti" Durum Makinesi

```
pkptr = op_pk_create_fmt ("ss_baglanti_istek_pk");
op_pk_nfd_set (pkptr, "KaynakID", TermID);
su_an = op_sim_time();
kullanilan_slot = (int) floor ((su_an / slot_genisligi) + EPSILON);
simdiki_slot = kullanilan_slot % slot_sayisi;
/* Paketi gondermek icin yeterli zaman var mi?*/
kalan_zaman = ((kullanilan_slot + 1)*slot_genisligi) - su_an;
pk_genisligi = (double) op_pk_total_size_get (pkptr);
pk_zamani = (double) pk_genisligi / tx_data_rate;
```

```
if((pk_zamani<kalan_zaman)&&!Busy)      op_pk_send (pkptr, TX_OUT_STRM);
else     op_intrpt_schedule_self(op_sim_time()+(1*slot_genisligi),10);
```

“Ek_Slot” Durum Makinesi

```
pkptr = op_pk_create_fmt ("ss_burst_istek_pk");
op_pk_nfd_set (pkptr, "KaynakID", TermID);
su_an = op_sim_time();
kullanilan_slot = (int) floor ((su_an / slot_genisligi) + EPSILON);
simdiki_slot = kullanilan_slot % slot_sayisi;
kalan_zaman = ((kullanilan_slot + 1)*slot_genisligi) - su_an;
pk_genisligi = (double) op_pk_total_size_get (pkptr);
pk_zamani = (double) pk_genisligi / tx_data_rate;
if((pk_zamani<kalan_zaman)&&!Busy)      op_pk_send (pkptr, TX_OUT_STRM);
else     op_intrpt_schedule_self((op_sim_time()+(1*slot_genisligi)),40);
```

“Paket_Al” Durum Makinesi

```
if((pkptr = op_pk_get (RX_IN_STRM))!=OPC_NIL)
        {op_pk_nfd_get (pkptr, "HedefID", &Hedef);
                if(Hedef==TermID)
                {op_pk_format(pkptr,pk_format);
                if ((strcmp("ss_baglanti_cevap_pk",pk_format)==0) ||
                (strcmp("ss_burst_cevap_pk",pk_format)==0))
                        {op_pk_nfd_get (pkptr, "DataSlot", &dataSlot);
                        replay=OPC_TRUE;}
                        else if ((strcmp("ss_slot_iptal_pk",pk_format)==0))
                                {op_pk_nfd_get (pkptr, "DataSlot", &dataSlot);
                                slotiptal=OPC_TRUE;
                                replay=OPC_TRUE;}}
                op_pk_destroy(pkptr);
        }
```

“Slot_Ayar” Durum Makinesi

```
replay=OPC_FALSE;
if (slotiptal==OPC_TRUE)
        {slotiptal=OPC_FALSE;
        for(i=0;i<eleman;i++)
        {if(benim_slot[i]==dataSlot) //dizinin elemani
                {for (k=i;k<eleman;k++)
                if(benim_slot[k+1]==-1)      {benim_slot[k]=-1;break;}
                else                         benim_slot[k]=benim_slot[k+1];
                eleman--;
                break;
        }//if}//for}//içte if
else    {benim_slot[eleman]=dataSlot; eleman++;}//slot iptal edilecek...
```

“Uyku” Durum Makinesinin Enter Execs bölümü

```
if (!paketGeldi)
        {uykusay++;
        op_stat_write(uykusayist,uykusay);
        op_intrpt_schedule_self (op_sim_time ()+UyumaSure,60);//sleep mode icin
        }
paketGeldi=OPC_FALSE;
```

“Uyku” Durum Makinesinin Exit Execs bölümü

```
if(Uyku_Periyot_Sonu)//sleep mode end intrpt ise
        op_intrpt_schedule_self (op_sim_time ()+UyumaSure,50);
else
        {
        paketal();
        paketGeldi=OPC_TRUE;
        }
```

“Gönder” Durum Makinesi

```
if (DATA_ENQ)
{
        su_an = op_sim_time();
        kullanilan_slot = (int) floor ((su_an / slot_genisligi));
        simdiki_slot = kullanilan_slot % slot_sayisi;
        kalan_zaman = ((kullanilan_slot + 1)*slot_genisligi) - su_an;
        pk_genisligi = (double) op_pk_total_size_get (op_subq_pk_access (0,
        OPC_QPOS_HEAD));
        pk_zamani = (double) pk_genisligi / tx_data_rate;
        buldu=-1;slotayar=1;
        for (i=0;i<dboy;i++)
        {if(benim_slot[i]==-1)
        break;
        else if ((simdiki_slot==benim_slot[i])&& (pk_zamani<kalan_zaman))
        {pkptr = op_subq_pk_remove (0, OPC_QPOS_HEAD);
        op_pk_send (pkptr, TX_OUT_STRM);
        buldu=i;
        break;}
        }//for sonu
        if((UykuModu==1)&&(!Busy)&&(DATA_ENQ))
        {pkptr = op_subq_pk_remove (0, OPC_QPOS_HEAD);
        op_pk_send (pkptr, TX_OUT_STRM);}
//Eger gonderilecek veri varsa ve terminale ait yeni bir slot varsa
//birsonraki slot degerine interrupt kurmak gerekiyor.
        if (DATA_ENQ)
        {if((buldu!=-1)&&(benim_slot[buldu+1]!=-1))
                if (benim_slot[buldu+1]>=benim_slot[buldu])
                        slotayar=benim_slot[buldu+1]-simdiki_slot;
```

```
                else
                        slotayar=(slot_sayisi-simdiki_slot)+benim_slot[buldu+1];
                else if((buldu!=-1)&&(buldu==eleman-1))
                        slotayar=(slot_sayisi-simdiki_slot)+benim_slot[0];}
op_intrpt_schedule_self (((kullanilan_slot+slotayar)*slot_genisligi+EPSILON ),0);}
```

“Parçala” Durum Makinesi

paketal();//kaynaktan gelen paketleri parçalayıp kuyruğa atan fonksiyon.

“Bekle” Durum Makinesi

current_intrpt_type = op_intrpt_type ();//gelen interrupın tipine bakar.

Kullanılan Foksiyonlar

```
static void paketal()
        {int i,hucre_sayisi;
        Packet *pkptr;
        FIN (paketal());
        if(((pkptr =  op_pk_get (SRC_IN_STRM))!=OPC_NIL))
        {
        hucre_sayisi=(int) floor (op_pk_total_size_get(pkptr)/(50*8*2));
                if(hucre_sayisi<1)
                        hucre_sayisi=1;
                for(i=1;i<hucre_sayisi+1;i++)
                {
                pkptr=op_pk_create_fmt("ss_data_pk");
                op_pk_nfd_set (pkptr, "KaynakID",TermID);
                op_subq_pk_insert (0, pkptr, OPC_QPOS_TAIL);
                }
        if ((op_subq_stat (0,
OPC_QSTAT_BITSIZE)>max_queue)&&(UykuModu==1))
        {op_intrpt_schedule_self(op_sim_time(),40);
        //ciftslot==OPC_TRUE;}
        }
        FOUT;
        }
```

Durum Makineleri Arası Geçiş Değerleri ve Bazı Sabit Değerler

```
#include <math.h>
#define EPSILON             0.000000000001
#define dboy                (5)
#define slot_sayisi         (120)
#define UyumaSure           (50)
#define SRC_IN_STRM         (0)
#define TX_OUT_STRM         (1)
#define SINK_OUT_STRM       (0)
```

```
#define RX_IN_STRM          (1)
#define max_queue           (15000)
#define DATA_ENQ            (!(op_subq_empty(0)))
#define Kaynak_Paket        (current_intrpt_type == OPC_INTRPT_STRM) &&
                            (op_intrpt_strm () == SRC_IN_STRM)
#define RX_Paket            (current_intrpt_type == OPC_INTRPT_STRM) &&
                            (op_intrpt_strm () == RX_IN_STRM)
#define Slot                (current_intrpt_type == OPC_INTRPT_SELF) &&
                            ((op_intrpt_code () == 0))
#define Uyku_Periyot        (op_intrpt_type() == OPC_INTRPT_SELF) &&
                            (op_intrpt_code () == 50)
#define Uyku_Periyot_Sonu   (op_intrpt_type() == OPC_INTRPT_SELF) &&
                            (op_intrpt_code () == 60)
#define Busy                (op_stat_local_read (0) == 1.0)
#define Istek_Cevap         (replay == OPC_TRUE)
#define Baglanti_Istek      (op_intrpt_type() == OPC_INTRPT_SELF) &&
                            (op_intrpt_code () == 10)
#define Ek_Slot_Istek       (op_intrpt_type() == OPC_INTRPT_SELF) &&
                            (op_intrpt_code () == 40)
```

EK - B : EDSMAC PROTOKOLÜ MD'nin PROSES MODELİNE AİT PROGRAM KODLARI

Bu ekte, EDSMAC protokolünün MD için kullandığı proses modeline ait C program kodları verilmektedir.

"Basla" Durum Makinesi

```
brequest=0;
connection=OPC_FALSE;
data=0;
op_prg_mem_free(&st);
st = (slot_tablo*) op_prg_mem_alloc (sizeof(slot_tablo));
for(i=0;i<slot_sayisi;i++)
        {st->term[i]=-1; st->oncelik[i]=0;}
```

"Paket_Al" Durum Makinesi

```
if((pkptr = op_pk_get (RX_IN_STRM))!=OPC_NIL)
        {
                op_pk_format(pkptr,pk_format);
                op_pk_nfd_get (pkptr, "KaynakID", &TermID);
                if(strcmp("ss_data_pk",pk_format)==0)
                        {data=1; brequest=0; }
                else if (strcmp("ss_baglanti_istek_pk",pk_format)==0)
                        {brequest=1; connection=OPC_TRUE; data=0; }
                else if (strcmp("ss_burst_istek_pk",pk_format)==0)
                        {brequest=1; data=0; }
                else    {op_pk_destroy(pkptr); }
        }
```

"Veri" Durum Makinesi

```
if(TermID==1)        op_pk_send(pkptr,SINK1_OUT_STRM);
else if(TermID==2)   op_pk_send(pkptr,SINK2_OUT_STRM);
else                 op_pk_destroy(pkptr);
```

"Bekle" Durum Makinesi

```
//Bu bölümde olay meydana gelene kadar bekliyor.
current_intrpt_type = op_intrpt_type ();//gelen interrupın tipini alır.
```

“Slot_Tahsis” Durum Makinesi

```
if (connection)//yeni bir terminal geldiginde baglanti yapilip cevap paketi yollaniyor.
        {connection=OPC_FALSE;
        for(i=1;i<slot_sayisi;i++)
                {if(st->term[i]==-1)
                        {st->term[i]=TermID;
                        st->oncelik[i]=1;
                        pkptr=op_pk_create_fmt("ss_baglanti_cevap_pk");
                        op_pk_nfd_set (pkptr, "HedefID",TermID);
                        op_pk_nfd_set (pkptr, "DataSlot",i);
                        op_pk_send(pkptr,TX_OUT_STRM);
                        break;}//if
                }//for
        }//if (connection)
else//eger connection true degilse baglanti istegi yoksa extraslot istegi var demektir.
        {for(i=extra_slot_bas;i<slot_sayisi;i++)
                {buldu=OPC_FALSE;
                if((st->term[i]==-1))
                        {buldu=OPC_TRUE;
                        st->term[i]=TermID;
                        st->oncelik[i]=2;
                        pkptr=op_pk_create_fmt("ss_burst_cevap_pk");
                        op_pk_nfd_set (pkptr, "HedefID",TermID);
                        op_pk_nfd_set (pkptr, "DataSlot",i);
                        op_pk_send(pkptr,TX_OUT_STRM);
                        break; }//if}//for
        if (buldu==OPC_FALSE)
                {i=extra_slot_bas+floor(op_dist_uniform(slot_sayisi-extra_slot_bas));
                eskiterm=st->term[i];
                pkptr=op_pk_create_fmt("ss_slot_iptal_pk");
                op_pk_nfd_set (pkptr, "HedefID",eskiterm);
                op_pk_nfd_set (pkptr, "DataSlot",i);
                op_pk_send(pkptr,TX_OUT_STRM);
                st->term[i]=TermID;
                st->oncelik[i]=2;
                pkptr=op_pk_create_fmt("ss_burst_cevap_pk");
                op_pk_nfd_set (pkptr, "HedefID",TermID);
                op_pk_nfd_set (pkptr, "DataSlot",i);
                op_pk_send(pkptr,TX_OUT_STRM);
                break;}//if buldu
}//else extra slot istegi
```

EK - C : DCF PROTOKOLÜ AD'nin PROSES MODELİNE AİT PROGRAM KODLARI

Bu ekte, DCF protokolünün AD için kullandığı proses modeline ait C program kodları verilmektedir.

"Basla" Durum Makinesi

```
my_id = op_id_self();
my_node_id = op_topo_parent (my_id);
op_ima_obj_attr_get (my_node_id, "TerminalID", &terminalID);
op_ima_obj_attr_get (my_node_id, "HedefID", &destinationID);
op_ima_obj_attr_get (my_node_id, "Uretec.Start Time", &baslama);
op_ima_obj_attr_get (my_node_id, "Uretec.Stop Time", &bitis);
op_ima_obj_attr_get (my_node_id, "Uyku", &uyusunmu);
/* Transmitter veri hizini al */
tx_id =  op_topo_assoc (my_id, OPC_TOPO_ASSOC_OUT, OPC_OBJTYPE_RATX, 0);
comp_id = op_topo_child (tx_id, OPC_OBJTYPE_COMP, 0);
tx_ch_id = op_topo_child (comp_id, OPC_OBJTYPE_RATXCH, 0);
op_ima_obj_attr_get (tx_ch_id, "data rate", &tx_data_rate);
if (uyusunmu==0)
        op_intrpt_schedule_self (op_sim_time ()+SleepTime, Uyku);
slotTime = 20E-06;
farkli_int=OPC_FALSE;uykupaket=OPC_FALSE;
/* Short interframe gap in terms of seconds.*/
sifsTime = 10E-06;
/* PLCP overheads, which include the preamble and header, in     */
/* terms of seconds.*/
plcp_overhead_control = 192E-06;
plcp_overhead_data    = 192E-06;
/* Minimum contention window size for selecting backoff slots.   */
cwMin = 31;
/* Maximum contention window size for selecting backoff slots.   */
cwMax = 1023;
difsTime = sifsTime + 2 * slotTime;
//collision durumunda deðeri artirilir
retryCount = 0;
backoffRequired=OPC_FALSE;
paketGeldi=OPC_FALSE;
backoffSlots=0;
backoffSlotsCount=0;
maxBackoff=cwMin;
navDuration = 0;
```

```
copypkptr=OPC_NIL;
paketBasariliGonderildi=OPC_FALSE;
sentPacketCount=0;
recivedPacketCount=0;
yeniden=OPC_FALSE;
ackEnd=0;
paketGeldi=OPC_FALSE;
recivedACKPacketCount=0;
ackEndEvhCancelCount=0;
//istatistiksel bilgiler...
dfs=0;
sentPacketCountStat = op_stat_reg("Gonderilen Paket",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
recivedPacketCountStat = op_stat_reg("Alinan Paket",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
queuePacketCountStat = op_stat_reg("Kuyruk Paket Sayisi",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
recivedACKPacketCountStat = op_stat_reg("Alinan ACK Paket",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
ackEndStat = op_stat_reg("ACK End Sayisi",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
ackEndEvhCancelCountStat = op_stat_reg("ACK Iptal Sayisi",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
backoffStat = op_stat_reg("Backoff Slots Degeri",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
backoffCountStat = op_stat_reg("Backoff Sayisi",
                        OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
dfsist = op_stat_reg("dfs", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
op_intrpt_schedule_self (op_sim_time (), 0);
```

“Kaynak” Durum Makinesi

```
ustKatmandanPaketAl();
        if (op_ev_valid (difsEndEvh) == OPC_FALSE)
                if(!Busy)
                        difsEndEvh=op_intrpt_schedule_self (op_sim_time ()
                        +difsTime, DIFS_END);
                else
                        backoffRequired=OPC_TRUE;
```

“DIFS” Durum Makinesi

```
if(!Busy)
{if (op_ev_valid (difsEndEvh)==OPC_FALSE)
        difsEndEvh=op_intrpt_schedule_self (op_sim_time ()+
        difsTime+PRECISION_RECOVERY, DIFS_END);
else
        {op_ev_cancel (difsEndEvh);backoffRequired=OPC_TRUE;}}
```

“SIFS” Durum Makinesi

```
pkptr=op_pk_create_fmt("cc_CSMA_ACK_pk");
op_pk_nfd_set (pkptr, "destination",destinationID);
op_pk_send(pkptr,TX_OUT_STRM);
```

“ACK” Durum Makinesi

```
ackEnd++;
op_stat_write(ackEndStat, ackEnd);
backoffRequired=OPC_FALSE;
backoffSlots=0;
retryCount++;
```

“UykuModu” Durum Makinesi

```
if(!uykupaket)
{op_intrpt_schedule_self (op_sim_time ()+SleepTime, UykuSonu);}
uykupaket=OPC_FALSE;
//*****Exit Execs*****
current_intrpt_type = op_intrpt_type ();
if(SleepEnd)
        {op_intrpt_schedule_self (op_sim_time ()+SleepTime, Uyku);}
else if(FromSrc)
        {ustKatmandanPaketAl();uykupaket=OPC_TRUE;}
else
        uykupaket=OPC_TRUE;
```

“Backoff” Durum Makinesi

```
if(!paketGeldi)
{if (backoffSlots==0)
{  if(retryCount==0)
                maxBackoff=cwMin;
        else
                maxBackoff=maxBackoff*2+1;
        if(maxBackoff>cwMax)
                maxBackoff=cwMax;
        // +1 ifadesi yazilmadiginda self int zaman hatasi veriyor
        backoffSlots=1+floor(op_dist_uniform(maxBackoff+1));
}
op_stat_write(backoffStat, backoffSlots);
backoffSlotsCount++;
op_stat_write(backoffCountStat, backoffSlotsCount);
contentionStart=op_sim_time();
backoffEndEvh =
op_intrpt_schedule_self(op_sim_time()+backoffSlots*slotTime,BACKOFF_END);}
paketGeldi=OPC_FALSE;
```

```
//****Exit Execs***
current_intrpt_type = op_intrpt_type ();
if(ChannelStat || FromRx)
        backoffSlots -=  ceil ((op_sim_time()-contentionStart -
PRECISION_RECOVERY) / slotTime);
if (op_ev_valid (backoffEndEvh)&&op_ev_pending(backoffEndEvh))
        op_ev_cancel (backoffEndEvh);
if(FromSrc)
        {ustKatmandanPaketAl();paketGeldi=OPC_TRUE;}
```

“Gonder” Durum Makinesi

```
if (DataENQ)
{if (!tekrarpaket)
        {       pkptr = op_subq_pk_remove(0, OPC_QPOS_HEAD);
                zaman=op_pk_creation_time_get(pkptr);
                if (ID==-1)
                {ID=floor(op_dist_uniform(1000000)+1);}
                op_pk_nfd_set (pkptr, "ID",ID);
                op_pk_nfd_set (pkptr, "time",zaman);
                ackTime=((double)op_pk_total_size_get(pkptr)/tx_data_rate);
                op_pk_send (pkptr, TX_OUT_STRM);}
        else{   pkptrtekrar=op_pk_create_fmt("cc_CSMA_data_pk");
                op_pk_nfd_set (pkptrtekrar, "source",terminalID);
                op_pk_nfd_set (pkptrtekrar, "destination",destinationID);
                op_pk_nfd_set (pkptrtekrar, "ID",ID);
                op_pk_nfd_set (pkptrtekrar, "time",zaman);
                ackTime=((double)op_pk_total_size_get(pkptrtekrar)/tx_data_rate);
                op_pk_send (pkptrtekrar, TX_OUT_STRM);}
        sentPacketCount++;
        backoffSlots=0;
        backoffRequired=OPC_FALSE;
        op_stat_write(sentPacketCountStat, sentPacketCount);
        ackEndEvh =
op_intrpt_schedule_self(op_sim_time()+3*ackTime,ACK_END);}
ackGeldi=OPC_FALSE;
paketGeldi=OPC_FALSE;
```

“PaketAl” Durum Makinesi

```
if((pkptr = op_pk_get (RX_IN_STRM))!=OPC_NIL)
        {op_pk_nfd_get (pkptr, "destination", &destID);
                if(destID==terminalID)
                {op_pk_format(pkptr,pk_format);
                if (strcmp("cc_CSMA_data_pk",pk_format)==0)
                        {op_pk_send(pkptr,SINK_OUT_STRM);
                        recivedPacketCount++;
                        op_stat_write(recivedPacketCountStat, recivedPacketCount);
```

```
        op_intrpt_schedule_self(op_sim_time()+sifsTime,SIFS_END);}
                else if (strcmp("cc_CSMA_ACK_pk",pk_format)==0)
                        {op_pk_destroy(pkptr);ackGeldi=OPC_TRUE;
                         tekrarpaket=OPC_FALSE;
                         op_subq_pk_remove(0, OPC_QPOS_HEAD);
                         ID=-1;
                        if (op_ev_valid (ackEndEvh) )
                                {op_ev_cancel (ackEndEvh);
                                ackEndEvhCancelCount++;
                                op_stat_write(ackEndEvhCancelCountStat,
ackEndEvhCancelCount);}
                                packetSending=OPC_FALSE;
                                backoffRequired=OPC_FALSE;
                                backoffSlots=0;
                                retryCount=0;
                                recivedACKPacketCount++;
                                op_stat_write(recivedACKPacketCountStat,
recivedACKPacketCount);}
                        }else op_pk_destroy(pkptr);
        }//if
paketGeldi=OPC_FALSE;
```

“Bekle” Durum Makinesi

current_intrpt_type = op_intrpt_type ();//gelen interrupın tipini alır.

“Ertele” Durum Makinesi

backoffRequired=OPC_TRUE;//difs sonunda ortam meşgulse backof var

ÖZGEÇMİŞ

1978 yılında İzmit'te doğdu. İlk, orta ve lise öğrenimini İzmit'te tamamladı. 1996 yılında girdiği Kocaeli Üniversitesi Bilgisayar Bilimleri Mühendisliğinden 2002 yılında Bilgisayar Müdendisi olarak mezun oldu. 2002 yılından itibaren Kocaeli Üniversitesi Enformatik Bölümünde Öğretim görevlisi olarak çalışmaktadır. 2005 yılında başladığı Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'ndaki Yüksek Lisans eğitimine devam etmektedir.