KABLOSUZ ALGILAYICI AĞLARDA ENERJİ VERİMLİ MAC PROTOKOLÜ TASARIMI VE UYGULAMASI

Sinan TOKLU

DOKTORA TEZİ
ELEKTRONİK – BİLGİSAYAR EĞİTİMİ

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

Şubat 2013
ANKARA

Sinan TOKLU tarafından hazırlanan KABLOSUZ ALGILAYICI AĞLARDA ENERJİ VERİMLİ MAC PROTOKOLÜ TASARIMI VE UYGULAMASI adlı bu tezin Doktora tezi olarak uygun olduğunu onaylarım.

Doç. Dr. O.Ayhan ERDEM
Tez Yöneticisi

Bu çalışma, jürimiz tarafından oy birliği ile Elektronik-Bilgisayar Eğitimi Anabilim Dalında Doktora tezi olarak kabul edilmiştir.

Başkan: : Prof. Dr. İsmail ERTÜRK

Üye : Prof. Dr. Ömer Faruk BAY

Üye : Prof. Dr. M. Ali AKCAYOL

Üye : Doç. Dr. O.Ayhan ERDEM

Üye : Doç. Dr. Erdoğan DOĞDU

Tarih : 18/02/2013

Bu tez, Gazi Üniversitesi Bilişim Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

KABLOSUZ ALGILAYICI AĞLARDA ENERJİ VERİMLİ MAC PROTOKOLÜ TASARIMI VE UYGULAMASI
(Doktora Tezi)

Sinan TOKLU

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

ŞUBAT 2013

ÖZET

Kablosuz algılayıcı ağlar (KAA) günümüzde birçok alanda karşımıza çıkan bir teknolojidir ve önemi de gün geçtikçe artmaktadır. Bu ağlar, kablosuz algılayıcı ağlarda kullanılan çok sayıda dağıtılabilen ve kendi kendini organize edebilen düğümler içermektedir. Algılayıcı düğümler, genellikle ağda kolayca yerleştirilmeleri için pil ile çalıştırılmaktadırlar. Özellikle de ulaşılması zor olan bölgelerdeki KAA için enerji verimliliği çok kritik bir öneme sahip olmaktadır. Dolayısıyla KAA için en kritik konu haline gelen enerji verimliliği için bu çalışmada düğümlerin uyarlamalı olarak uyuması, iletişimin devamlılığının sağlanması ve düğümlerin enerjilerini en iyi şekilde kullanabilmeleri için yeni bir çalışma yapılmıştır. Bu çalışmada enerji israfının en fazla olduğu durumlar tespit edilmiştir. Bu durumlar için çözüm önerileri sunulmuş ve bu çözüm önerilerinin hem benzetimi hem de uygulaması gerçekleştirilmiştir. Bu çalışmada enerji tasarrufu enerji israfının fazla olduğu durumları önlemek için, boşta kalan düğümlerin uyku durumuna geçirilmesi benimsenmiştir. Düğümlerin uyku durumuna geçiş işlemi uyarlamalı olarak yapılmaktadır. BSC-MAC(Base Station Controlled Medium Access Control-Baz İstasyon Kontrollü Ortam Erişim Kontrolü) protokolünde iki tür iletişim gerçekleştirilmektedir. Bunlar kök ve kaynak düğüm olarak ayrılmaktadır.

Kaynak düğümler, verileri üretip kök düğümler üzerinden baz istasyonuna iletme işlemini gerçekleştirirler. Hem kök düğümlerin hem de kaynak düğümlerin kendi aralarındaki iletişimin aksamaması için uyuma periyotlarının zaman senkronizasyonu uyarlamalı olarak iki farklı yöntemle gerçekleştirilmiştir. Elde edilen sonuçlara bakıldığında BSC-MAC protokolünün diğer var olan protokollerden daha iyi enerji verimliliği sağladığı, iletimin daha sağlıklı ve düğümler arasında adaletli erişim hakkının daha iyi olduğu görülmüştür.

Bilim Kodu : **702.1.014**
Anahtar Kelime : **kablosuz algılayıcı ağlar, kablosuz algılayıcı düğümler, enerji-verimliliği, IEEE 802.15.4, MAC protokolleri**
Sayfa Adedi : **125**
Tez Yöneticisi : **Doç. Dr. O. Ayhan ERDEM**

DESIGN AND IMPLEMENTATION OF AN ENERGY EFFICIENT MAC PROTOCOL FOR WIRELESS SENSOR NETWORKS

(Ph.D. Thesis)

Sinan TOKLU

GAZİ UNIVERSITY
INFORMATION INSTITUTE

February 2013

ABSTRACT

Wireless Sensor Network is a popular technology in the different fields nowadays and is getting more important day after day. This network contains nodes which is used to in wireless sensor network and is organized oneself. Sensor nodes generally are worked with a battery to be placed easily in the network. So, energy effeciency hold especially so critic significance for region where is reached difficultly. Therefore, this new study covers nodes adaptive sleeping, providing constant transmission, and using nodes energy efficiently for energy effeciency which is so important issue of wireless sensor network. The highest energy wastages status is determined in this study. This study offers solutions for that status and practiced both of simulation and application. This study embraces that fee nodes are passed in the sleeping status to prevent the highest energy wastage status. Application for transferring sleeping status in the nodes is made adaptively. Two type transmissions are achieved in the BSC-MAC protocol. They are seperated as root and resource. Resource nodes produce data and convey it through root nodes to base stations. Sleeping periot's time synchronizing is achieved with two different methods adaptively to

prevent failing transmission between root and resource or in themself. As a result, BSC-MAC protocol is more efficient method than others in terms of energy efficiency and also transmission in the BSC-MAC protocol is well. Also BSC-MAC protocol achieves a fair access to the media.

Science Code : **702.1.014**
Key Words : **wireless sensor networks, wireless sensor nodes, energy-efficient, IEEE 802.15.4, MAC protocols**
Page Number : **125**
Adviser : **Assoc. Prof. Dr. O. Ayhan ERDEM**

TEŞEKKÜR

Tez çalışmam süresince, bilgi ve önerileri ile katkıda bulunan, çalışmalarımı yönlendiren ve doktoramın her aşamasında desteğini esirgemeyen Hocam Sayın Doç. Dr. O. Ayhan ERDEM'e teşekkürlerimi sunarım.

Çalışmalarıma kıymetli bilgi ve önerileri ile katkıda bulunan tez izleme komitesi üyeleri Sayın Prof. Dr. İsmail ERTÜRK ve Prof. Dr. Ömer Faruk BAY'a, ayrıca Dr. Mehmet ŞİMŞEK'e kıymetli bilgi ve önerilerinden dolayı teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

ÇİZELGELER LİSTESİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı kısaltmalar açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklama**
λ	Gelen paket sayısı
h	Düğümden geçiş için gereken zaman
α	Seviye

Kısaltmalar	**Açıklama**
KAA	Kablosuz Algılayıcı Ağlar
MAC	Medium Access Control Ortam Erişim Kontrolü
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance Çakışma Koruması ile Taşıyıcı Duyarlı Çoklu Erişim
CSMA-CD	Carrier Sense Multiple Access wirh Collision Avoidance Çakışma Karşılaşmalı Taşıyıcı Duyarlı Çoklu Erişim
LR-WPAN	Low-Rate Wireless Personal Area Networks Düşük Puan Kablosuz Kişisel Alan Ağları
RTS	Request To Send-Gönderme İsteği Sinyali
CTS	Clear To Send-Gönderebilme İzin Sinyali
PAN	Personal Area Network-Kişisel Alan Ağı
ACK	Acknowledgement-Alındı Bilgisi
TDMA	Time Division Multiple Access Zaman Bölmeli Çoklu Erişim
MCU	Micro Controller Unit-Mikro Denetleyici Birimi

DAPC	Robust Distributed Active Power Control Sağlam Dağıtık Aktif Güç Kontrolü
FRSM	Future Request To Send Message Mesaj Gönderimi İçin Gelecek Talep
TRAMA	Traffic-Adaptive Medium Access Uyarlamalı Trafik Ortam Erişimi
S-MAC	Sensor Medium Access Protocol Algılayıcı Ortam Erişim Protokolü
FLAMA	Flow-Aware Medium Access Protocol Akış Farkında Ortam Erişim Protokolü
PAMAS	The Power Aware Multiaccess With Signalling For Adhoc Networks Adhoc Ağlar İçin Sinyalizasyon ile Güç Farkında Çoklu Erişim
QFT	Quantative Feedback Theory-Nicel Geri Besleme Teorisi
CC-MAC	Spatial Correlation-Based Collaborative Medium Access Protocol Mekansal İlişki Tabanlı İşbirlikçi Ortam Erişim Protokolü
STEM	Sparse Topology and Energy Management Seyrek Topoloji ve Enerji Yönetimi
Z-MAC	Zebra Medium Access Control Zebra Ortam Erişim Kontrolü
CMAC	Convergent Medium Access Control Yakınsak Ortam Erişim Kontrolü
O-MAC	Organized Medium Access Control Organize Ortam Erişim Kontrolü
B-MAC	Berkeley Medium Access Control Berkeley Ortam Erişim Kontrol
AEEMAC	Adaptive Energy Efficient Medium Access Control Protocol for Wireless Sensor Networks Kablosuz Algılayıcı Ağlar için Uyarlamalı Enerji Verimli Ortam Erişim Kontrol Protokolü

FIFO	First In First Out-İlk Giren İlk Çıkar
P-MAC	Pattern Medium Access Control Desenli Ortam Erişim Kontrolü
T-MAC	Timeout Medium Access Control Zaman Aşımı Ortam Erişim Kontrolü
DSMAC	Medium Access Control With A Dynamic Duty Cycle For Sensor Networks Algılayıcı Ağlar İçin Dinamik Görev Çevrimli Ortam Erişim Kontrolü
TICER	Transmitter Initiated Cycled Receiver Çevrimsel Alıcıyı Başlatan İletici
RICER	Receiver Initiated Cycled Receiver Çevrimsel Alıcıyı Başlatan Alıcı
ER-MAC	Emergency Response Medium Access Control Acil Cevap Ortam Erişim Kontrolü
OTS	Order To Sleep - Uyku Talimatı
NTS	Node To Sleep - Uyuyacak Düğüm
BO	Beacon Order - İşaret Sinyal Komutu
STF	Super Time Frames - Süper Zaman Çerçeveleri
PRTF	Pattern Repeat Time Frame Desen Tekrarlama Zaman Çerçevesi

1. GİRİŞ

KAA akademik ve endüstriyel alanda çok sık olarak kullanılmaya ve önemi gün geçtikçe artmaya başlayan bir teknoloji olarak karşımıza çıkmaktadır. Algılayıcı düğümler bir ortama ya rastgele atılırlar ya da yerleri belirlenerek ilgili konumlara yerleştirilirler [1]. Özellikle ulaşılması zor veya tehlikeli olan bölgelere düğümlerin rastgele atılması ve bu düğümlerin kendi kendilerini organize edebilecek yeteneğe sahip olmalarından dolayı birçok alanda kullanılmaktadırlar. Özellikle askeri olaylarda, hayvan, bitki yaşamlarının izlenmesinde, hedef yolun bulunmasında, inşaatların izlenmesinde, telemetri, hasta izleme, endüstriyel otomasyon kontrolleri ve bazı önemli araştırmalarda kullanılmaktadırlar [2].

Algılayıcı ağlar, algılama, işlem, iletişim ve kendi kendine organize olma yeteneklerine sahip olan çok sayıda algılayıcı düğümlerden oluşmaktadır. Bu ağlar çoklu sıçramalı kablosuz ağlarda kullanılan çok sayıda dağıtılmış ve kendi kendini organize edebilen düğümler içermektedirler. Algılayıcı düğümler, genellikle ağda kolayca yerleştirilmeleri için pil ile çalıştırılmaktadırlar. Bu tip ağlarda yerleştirilen çok sayıdaki düğümlerin pillerini şarj etmek ya çok zordur ya da imkânsızdır. Bundan dolayı KAA için enerji verimliliği çok kritik bir öneme sahip olmaktadır [1,2].

Algılayıcı ağlardaki amaç, ağı oluşturan düğümlerin ucuz ve güç kaynaklarının verimli olarak kullanılmasıdır. Ağda bulunan düğümlerin bazıları veya tamamı pillerinin bitmesi nedeniyle veya konum değiştirme nedenleri gibi olaylardan dolayı ağdaki ömürlerini bitirirler. Bu da zamanla ağın ömrünü tüketmektedir[1]. Dolayısıyla pillerin değiştirilemeyeceği veya şarj edilemeyecek bir ortamda bulunan düğümlerden en iyi şekilde yararlanmak ve ağın ömrünü uzatmak için düğümlerin enerji kaynaklarının verimli bir şekilde kullanılması gerekmektedir. Algılayıcı düğümlerdeki enerji verimliliği için MAC protokolünde yapılan tasarımlar önem olarak ilk sırada karşımıza çıkmaktadır [3].

MAC protokollerinde ortaya çıkan enerji israfı genelde şu şekilde sıralandırılabilir;

- Ortamın gereksiz yere dinlenmesi
- Kontrol paketlerinin fazlalığı
- İstem dışı alım
- Paket çakışması

Enerji israfının genel olarak en fazla olduğu durum ise ortamın gereksiz yere dinlenmesinden dolayı ortaya çıkmaktadır. Algılayıcı düğümler komşularından birinden bir mesaj gelip gelmediğini anlamak için mutlaka yolu dinlemeleri gerekmektedir. Bunun içinde radyo alıcılarını açarlar ve herhangi bir mesaj olmasa bile radyo alıcıları açık kalır. Bunun sonucu olarak düğümün yaşam ömrü hızlı bir şekilde bitmekte ve aynı zamanda ağın ömrü düğümlerin yaşam süreleriyle doğru orantılı olarak azalmaktadır [2,4].

BSC-MAC protokolünde amaç düğümlerin uyarlamalı olarak uyumasını, iletişimin devamlılığının sağlanmasını ve düğümlerin enerjilerini en iyi şekilde kullanabilmeleridir. Enerji israfının en fazla olduğu durumlarda bunlara bir çözüm bulmak ve ağın ömrünü artırmak amaçlanmıştır. Ayrıca ağın ömrünün uzatılması esnasında düğümlerin adaletli bir şekilde ortama erişiminin sağlanması ve böylece düğümlerin enerjilerini en verimli şekilde kullanabilmeleri amaçlanmıştır. BSC-MAC protokolünde enerji tasarrufu, enerji israfının fazla olduğu durumları önlemek için boşta kalan düğümlerin uyku durumuna geçirilmesi şeklinde olmaktadır. Bu uyku durumuna geçiş işlemi uyarlamalı olarak yapılmaktadır.

BSC-MAC protokolünde iki tür iletişim gerçekleştirilmektedir. Bunlar kök ve kaynak düğüm olarak ayrılmaktadır. Kaynak düğümler verileri üretip kök düğümler üzerinden baz istasyonuna iletme işlemini gerçekleştirirler. Baz istasyonuna olan iletimleri ise kök düğümler üzerinden gerçekleşmektedir. Hem kök düğümlerin hem de kaynak düğümlerin kendi aralarındaki ve birbirleri ile olan iletişimin aksamaması için uyuma periyotlarının zaman senkronizasyonu uyarlamalı olarak iki farklı yöntemle gerçekleştirilmiştir. Ağda bulunan düğümlerin hangisinin kök hangisinin kaynak düğüm olarak çalışacağına baz istasyonu kendi üzerinde çalışan algoritması

ile karar vermektedir. Kök düğümlerin uyuma zamanlarının ayarlanması ve onların bilgilendirme işlemi baz istasyonu tarafından yapılırken kaynak düğümlerde yolu ele geçiren düğüm diğer düğümleri uyku kipine geçirerek bu işlemi gerçekleştirmektedir. Kök düğümlerin ve kaynak düğümlerin uyku planlarının başlangıçta baz istasyonu tarafından yapılmaları nedeniyle zaman senkronizasyonu için optimum eşleme gerçekleştirilmektedir. Bu sayede ağdaki tüm düğümler ne yapacağını bildiğinden dolayı enerji verimi, var olan protokollerden daha iyi olmaktadır.

Bu çalışma beş bölümden oluşturulmuştur. Birinci bölüm giriş bölümü olup, ikinci bölümde; bu alanda yapılmış olan çalışmalar anlatılmıştır. Üçüncü bölümde KAA anlatılmıştır. Dördüncü bölümde ise BSC-MAC protokolünün benzetiminin ve uygulamasının açıklandığı bölüm olup, beşinci bölümde; sonuçlar verilmişitir.

2. YAPILMIŞ ÇALIŞMALAR

Burada enerji verimliliği için geliştirilmiş olan MAC protokolleri sunulacak daha sonra bu MAC protokollerine yapılan ek uygulamalar ve düğüm üzerinde uygulanan IEEE 802.15.4 ile CSMA(Carrier Sense Multiple Access-Taşıyıcı Duyarlı Çoklu Erişim) terimleri açıklanacaktır.

2.1. CSMA

CSMA protokolünde iletim işlemi ortamda başka iletim yapan yoksa başlatılmaktadır. CSMA protokolü kendine benzer protokollerden olan Aloha protokolüne göre daha fazla iş hacmi imkânı vermektedir. İletime geçilmeden önce hat dinlenir eğer hat meşgul değilse iletilir, meşgulse rastgele bir süre beklenir ve tekrar iletilir. Bu sayede oluşabilecek çarpışmalara engel olunmaya çalışılır [5]. Veri göndermek isteyen bir düğümün veya cihazın göndereceği veriyi göndermeden önce taşıyıcı dalga sinyalini dinlemesi veya beklemesi durumu olarak açıklanır. Taşıyıcı dalga, taşıma sinyali veya bilgi taşınmasına izin verme için gönderilen bir sinyal olarak kullanılmaktadır. Gönderici düğüm veri göndermeden önce başka bir düğümün sinyalinin hatta bulunup bulunmadığının belirlenmesi işlemini gerçekleştirir. Hatta o anda başka bir gönderim işlemi mevcutsa gönderici düğüm iletime başlamadan önce hattaki gönderimin tamamlanmasını beklemek durumundadır [6]. Bir iletim ortamının birden fazla düğüm tarafından kullanılması durumunu belirlemektedir.

CSMA Tipleri

1-Israrlı CSMA: Veri göndermek isteyen düğüm veri gönderme işlemine hazır olduğunda, devamlı olarak hattın meşgul olup olmadığını kontrol eder, eğer hatta bir işlem algılanmazsa küçük bir veri paketi gönderir. Bu gönderim esnasında bir çakışma gerçekleşirse, veri gönderme işlemi için rastgele bir zaman beklenir ve tekrar gönderim işlemi gerçekleştirilir [7].

p-Israrlı CSMA: Bu kural 1-Israrlı CSMA kuralının genelleştirilmiş hali olarak düşünebilir. Veri göndermek isteyen düğüm veri gönderme işlemine hazır olduğunda, daima hattın meşgul olup olmadığını kontrol eder. Hat meşgul değilse küçük bir veri paketi ve olabilirlik sinyalini beraber gönderir. Veri gönderme işlemi bir hata sonucu gerçekleşmezse, veriyi gönderen düğüm kendi için belirlenmiş bir sonraki özel zaman aralığında küçük veri paketini ve p sinyalini tekrar gönderir [6,7].

Israrlı olmayan CSMA: Veri göndermek isteyen düğüm veri gönderme işlemine hazır olduğunda, daima hattın meşgul olup olmadığını kontrol eder. Hat meşgul değilse veriyi göndermek isteyen veri gönderim işlemini başlatır. Hat meşgulse rastgele bir zaman beklenir ve veri gönderim işlemi tekrarlanır [6,7].

Israrlı CSMA kuralında hat meşgul değilken küçük bir veri paketi deneme amaçlı gönderilmekte iken, Israrlı olmayan CSMA kuralında direk veri gönderimine başlanmaktadır.

CSMA/CD(Carrier Sense Multiple Access wirh Collision Avoidance-Çakışma Karşılaşmalı Taşıyıcı Duyarlı Çoklu Erişim): Bu protokol, ağlar tarafından iletişim hatlarına erişim için kullanılan bir kontrol mekanizmasıdır. Ağda bulunan bir düğüm iletişim hattını veri göndermek üzere kullanacağı zaman hattın başka bir düğüm tarafından kullanılıp kullanılmadığına bakar. Ağda bulunan iki düğüm birbirlerine aynı anda veri gönderebilirler. Bu durumda gönderilen paketler arasında çarpışma olayı ortaya çıkacak ve hiçbir düğüm veri alma işlemini gerçekleştiremeyecektir. Ağda böyle bir durumun ortaya çıkması olayına çarpışma yakalama denilmektedir. Ağda böyle bir olay ortaya çıktığında ağ içerisinde bir sinyal üretilir ve ağ trafiğe belirli bir süre kapatılır. Ağda kullanılan hat düzeldikten sonra verileri gönderemeyen düğümler rastgele bir süre beklerler ve tekrar iletim gerçekleştirirler. Ağda oluşabilecek böyle olaylar ortalama milisaniyeler içerisinde gerçekleşmektedir [6].

CSMA/CA (Carrier Sense Multiple Access wirh Collision Avoidance-Çakışma Kaçınması ile Taşıyıcı Duyarlı Çoklu Erişim): CSMA/CD mekanizmasının

değiştirilmiş hali olan CSMA/CA kablosuz ağlarda kullanılmaktadır. Ağda bulunan tüm düğümler aynı frekansta iletim gerçekleştirirler. Kablosuz ağlarda iletim sırasındaki işaret gücü düşüktür bu yüzden mesafe kısa olmaktadır. Bu protokolde ağda bulunan tüm düğümlerin aynı bilgiyi alması olasılığı yerine veri iletiminden önce iki uç arasında kısa bir iletişim gerçekleştirilmektedir. Bu olayın avantajı ise ağda bulunan herhangi başka bir bilgisayar, hattın meşgul olduğunu bu iki uç sayesinde öğrenecek ve böylece iletişim gerçekleştirmeyecektir. Gönderilen kontrol mesajlarının çakışması durumu gerçekleşebilmektedir [7].

2.2. IEEE 802.15.4

IEEE 802.15.4 protokolü LR-WPAN'lar(Low-Rate Wireless Personal Area Networks-Düşük Yoğunluklu Kablosuz Kişisel Alan Ağları) için yeni bir standart olarak tasarlanmıştır. Bu protokolün hedefi ise çok az karmaşıklık, maliyet ve güçtür. Bunu düşük veri oranı olan kablosuz bağlanılabilirlik için yani pahalı olmayan sabit, portatif ve hareket eden cihazlar için hedeflemiştir. Bu standart ayrıca çoklu-sekmeli paket iletimini de desteklemektedir. 802.11'e göre KAA için daha uygundur.

Burada uyuma zamanı tasarımı için birçok özellik vardır. KAA'ların performansı bu özelliklerden etkilenmektedir[8]. Bu protokol CSMA/CA tabanlıdır, ancak bu protokolde RTS(Request To Send-Gönderme İsteği Sinyali)/CTS(Clear To Send-Gönderebilme İzin Sinyali) kontrol mesajları kullanılmamıştır. 802.11 gibi 802.15.4'te merkezileştirilmiş topolojileri desteklemektedir. Her 802.15.4 ağında tek bir cihaz PAN(Personel Area Networks-Kişisel Alan Ağları) koordinatörü olarak görev üstlenir ve ağdaki cihazların ilişkisini kontrol eder [8].

Bu protokol yıldız veya uç birimden uç birime topolojilerini kullanmaktadır. Yıldız topolojide bütün iletişim ve kaynak rezervasyonu PAN koordinatörü aracılığıyla olmaktadır. Düğümler uç birimden uç birime modelinde bağımsız olarak ve bir PAN koordinatör olmadan iletişim yapabilirler, ancak hepsi ağda öncelikli paylaşımcılık için PAN koordinatör ile ilişki kurmak zorundadır [9]. IEEE 802.15.4 MAC

protokolü genel olarak yıldız topoloji kullanmaktadır. Özellikleri bu topoloji için açık olarak belirlenmiş olmasına rağmen uç birimden uç birime topolojisi için açık değildir.

Bu protokol yıldız topoloji ile kullanıldığında 2 tane iletişim metodu kullanmaktadır. Bunlar işaret sinyali modu ve işaret sinyali olamayan mod olarak adlandırılmaktadır. İşaret sinyali modu seçiliyse iletişim bir ağ koordinatörü tarafından kontrol edilmektedir. Bu koordinatör cihaz eşlemesi için düzenli olarak işaret sinyali iletimi yapmaktadır. Ağ koordinatörü iletilen periyodik işaret sinyali tarafından çerçevenin başlangıç ve bitişini belirlemektedir. İşaret sinyali periyodunun uzunluğu ve sistemin görev çevrimi kullanıcı tarafından belirlenebilmektedir. Tabi ki kullanıcı bunları belirlerken standartta belirtilen limitler içerisinde bunu yapabilmektedir. İşaret sinyali modunda çerçeve aktif ve aktif olmayan periyotları içerir. Çerçevenin aktif parçası, 16 tane eşit zaman dilimi alanı içermekte ve 3 parçadan oluşmaktadır. İşaret sinyali CSMA kullanılmadan iletilir, zaman diliminin başlangıcı 0 olur. Koordinatör ise düğümlerle sadece aktif periyotta etkileşim halindedir ve aktif olmayan durumda uyku moduna geçebilir [8]. IEEE 802.15.4'te düşük gecikme işlemlerine izin vermek için garantili zaman dilimi opsiyonu bulunmaktadır. İşaret sinyali modunda ise güç-korunumu mekanizması varken, işaret sinyali olmayan modta güç korunumu mekanizması yoktur.

IEEE 802.15.4 Tabanlı KAA İçin Enerji Verimli MAC Tasarımı

Bu çalışmada IEEE 802.15.4 tabanlı kablosuz algılayıcı ağların performansını arttırmak için yeni bir enerji-verimli MAC tasarımı sunulmuştur. Önerilen mekanizma ağın trafik yüküne göre düğümlerin uyumalarına uyarlamalı olarak karar vermektedir. Bu işlem sayesinde iş hacmi ve enerji tüketiminde başarı elde edilmektedir. Bu mekanizma 2 tane aşama içermektedir. Bunlar değiş tokuş aşaması tasarımı ve oluşturma aşaması tasarımıdır. Değiş tokuş aşaması tasarımında, zamanlama parametreleri normal iletimin üstünde gerçekleşmektedir. Oluşturma aşaması tasarımında ise algılayıcı düğümlerin uyku zamanlarına tasarım

parametrelerinden faydalanılarak uyarlamalı olarak karar verilmektedir. Sonuçta, bütün algılayıcı düğümlerin tasarımları bir tasarıma doğru yakınsanmaktadır [9].

Kontrol aşırı yüklenmesini azaltmak için cihazların üstünde bulunan planlama parametreleri veri ve ACK paketlerinin içine gömülmüştür. Bundan dolayı ekstra paketlerden kaçınılmıştır. Ayrıca bir cihaz kendi planını ayarlamak için komşusunun planlama parametrelerini kullanmıştır. Bu mekanizma ile birçok plan yerine aşamalı olarak tek plana doğru gidilmektedir. Ağda oluşabilecek birçok plan enerji tüketimini önemli ölçüde arttıracaktır. Bu sayede bu işlem önlenmiştir[9].

IEEE 802.15.4 Uyumlu Cihazlar İçin Enerji-Verimli Plan

Bu çalışmada uyku modundaki güç tüketiminin analizi yapılmıştır. Özellikle indirme iletiminde dolaylı mod ve gönderme iletiminde direk modun analizi yapılmıştır. Araştırmacıların yaptığı araştırmalar sonucunda istem dışı alımın önemli enerji israfına neden olduğu ve kaçınılması gerektiği görülmüştür. IEEE 802,11'i kullanan WLAN'larda RTS/CTS mekanizması kullanılmaktadır ancak LR-WPAN'ların tasarımında RTS/CTS mekanizması kullanılmamaktadır. Kullanılmamasının nedeni ise sistemin karmaşıklığını azaltmaktır. İstem dışı alım problemine karşı bu çalışmada önerilen ise gönderme iletiminde geri çekilme mekanizması için önerilen bir uyku planlamasıdır [10].

IEEE 802.15.4 LR-WPAN'lar İçin Enerji Koruma Mekanizması

Bu çalışmada IEEE 802.15.4 LR-PAN'lar için enerji koruma mekanizması önerilmiştir. IEEE 802.15.4'ün önceki sürümünde her düğüm sadece çerçevenin aktif olmayan parçasında uyku moduna geçmektedir. Önerilen planda yönetici dinamik olarak görev döngüsüne karar vermekte ve BO(Beacon Order-İşaret Sinyal Komutu) çeşitliliği tarafından uyku modu aralığı ayarlanmaktadır. BO ise ağdaki gelen paketlerin oranı olarak kabul edilmiştir [11].

Koordinatör sonraki çerçeveyi oluşturduğunda, BO değeri çerçevenin uzunluğuna karar vermekte ve bu sayede o anki trafik durumuna göre ayarlanmaktadır. Düşük trafik yoğunluğunda BO değeri artmaktadır, bunun anlamı ise çerçevenin uzunluğu da artıyor demektir. Dolayısıyla uyku periyodu daha uzun olmaktadır. Bu işlem aktif periyot sabit uzunlukta olduğunda meydana gelmektedir. Buda düşük görev çevrimi elde etmeyi ve enerji tüketimini azaltmayı sağlamaktadır [11].

IEEE802.15.4 KAA İçin Güçlü Dağıtık Aktif Güç Kontrol Tekniği

Yeni pratik olarak uygulanabilir DAPC(Robust Distributed Active Power Control-Sağlam Dağıtık Aktif Güç Kontrolü) tekniği IEEE 802.15.4 için önerilmiştir. Bu teknikte QFT(Quantative Feedback Theory-Nicel Geri Besleme Teorisi) kullanılmıştır. Önerilen DAPC platformu alınan hedef sinyal güç göstergesi izleme tabanlıdır ve radyo kanallarının kararsızlığı ve algılayıcı düğümler arasındaki bozulmalara etki etmektedir [12]. Bu tekniğin anahtar özellikleri şu şekildedir;

1: Güç tüketimi ve zayiat olasılığı açısından ölçülebilir iyileşmeyi gerçekleştirmektedir.
2: Ağın çalışma şartlarındaki ilgili kesin bilgilerin elde edilmesi bu sayede radyo kanallarının kazanç sağlamaları ve kullanıcıların arasına girmeleri gerekmemektedir.
3: Önerilen grafiksel tasarım sistem performansının metriklerinin değiş tokuşunu basitleştirmektedir.

Çoklu Seviye İşaret Sinyalli 802.15.4 Algılayıcı Ağlar İçin Küme Yaşam Zamanını Eşitleme Algoritması

Çoklu-küme 802.15.4 algılayıcı ağlarında küme ve ağ yaşam süresini maksimize ederken öngörülmüş olay hassasiyetinin güvenirliliğini sürdürmedeki problemler dikkate alınmıştır. Kümeler köprüler üzerinden bağlanmışlardır. Bu köprüler aynı zamanda küme koordinatörü olarak çalışmaktadırlar. Bütün normal düğümler ve köprüler çakışmayı CSMA/CA algoritmasıyla çözmektedirler. Kümenin yaşam

süresi gereksiz düğümlerin kullanımı yolu ile maksimize edilmektedir ve bunların periyodik olarak uykuya gönderilmesi basit dağıtım aktivitesi yönetim algoritması ile gerçeklenmektedir. Ağın yaşam süresi bütün kümede düğümlerin ayarlanması yoluyla yani her geri çekilme periyodunda enerji tüketiminde eşitleme yaparak maksimize etmektedir. Bu problem analitik olarak modellenmiş ve veri sayfası kullanılarak yapılmıştır. Bunlar IEEE 802.15.4 kablosuz algılayıcı modülü tmote_sky ultra için yapılmıştır. Bu teknik sayesinde kümenin yaşam süresi eşitlenmiştir [13].

2.3. Algılayıcı Ağlar İçin Geliştirilen MAC Protokolleri

Algılayıcı ağlarda her düğüm kendi çevresindeki düğümleri organize edebilmektedir. Bu sayede iletişim belirli bir yoldan gerçekleşmektedir. MAC protokollerinde ise düğümlerin bağımsız ve en az karmaşıklıkla birbirleriyle iletişim yapması hedeflenmekte ve gerçekleştirilmektedir. Buradaki asıl amaç ise enerji verimliliği ve adaletli erişimdir. Bu protokollerde ortamın gereksiz yere dinlenmesinden dolayı ve paket çakışmalarından dolayı enerji kaybı ortaya çıkmaktadır. Birçok MAC protokolünde enerji verimliliği için genel olarak düğümlerin işleri olmadığı zaman uyku durumuna geçirilmektedir. Bu sayede düğümlerin uyku durumunda görev çevrimi az olacağından dolayı enerji tüketimi de azalmaktadır. Bazı protokoller uyku durumuna geçirmeden başka birde paketlerin öntakılarıyla oynayarak enerji verimliliği elde etmek istemişlerdir. Ayrıca yolu ele geçirme için kullanılan yöntemler TDMA(Time Division Multiple Access-Zaman Bölmeli Çoklu Erişim) ve CSMA üzerinde yapılan bazı değişiklikler ile enerji verimliliği sağlanmaya çalışılmıştır.

2.3.1. PAMAS

PAMAS (The Power Aware Multiaccess With Signalling-Sinyalizasyon ile Güç Farkında Çoklu Erişim) orijinal MAC protokolü ile ayrı bir sinyalleme kanalı kullanma düşüncesinin birleşimi olarak tasarlanmıştır. 2 tane alıcı vericiden yararlanarak enerjiyi korumaktadır. Bu 2 tane alıcı vericiden bir tanesi veri mesajları

için diğeri ise kontrol mesajları içindir. RTS ve CTS mesaj değiş tokuşu için paket gönderiminde kullanılan kanaldan ayrı olarak bir sinyalleme kanalı kullanılmaktadır. Bu ayrı sinyalleme kanalı düğümlere ne zaman ve ne kadar uzunlukta güçlerini kapatmaları için karar verme imkânı sunmaktadır. Bu protokolde bir düğüm 6 durumda olabilmektedir. Bunlar boşta, CTS bekleme, BEB(Binary Exponantial Backoff-İkili Üssel Geri Çekilme), paket bekleme, paket alma ve paket gönderme olarak adlandırılmaktadır [14-15].

Bir düğüm paket iletmiyor ve almıyorsa, iletecek paketi yoksa iletecek paketi varken komşusunun iletiminden dolayı iletemiyorsa boş modta bulunmaktadır. Düğüm iletecek bir paket aldığında RTS iletir ve CTS bekleme moduna geçer. Eğer beklenen CTS gelmez ise düğüm BEB moduna geçer. CTS gelirse paket iletmeye başlar ve paket gönderme moduna geçer. İstenilen alıcı, CTS iletimine bağlı olarak paket bekleme moduna geçer. Eğer paket bir gidiş geliş zamanı içinde gelmeye başlamaz ise, düğüm boş moda geçer. Eğer paket gelmeye başlar ise, sinyalleme kanalı üzerinde meşgul tonu iletir ve paket alma moduna geçer. Eğer hiçbir komşusu paket gönderme ve CTS bekleme durumunda değilse boş modtaki düğüm RTS aldığında, CTS ile karşılık verir. Bir düğüm için komşusunun paket gönderme modunda olduğunu bilmek kolaydır. Fakat RTS'nin iletimi komşu tarafından kontrol kanalı üzerinden başka iletimle çakışabileceğinden dolayı düğümün CTS bekleme modunda olduğunu bilmek her zaman mümkün değildir [15].

PAMAS ayrıca meşgul tonunu iletişimde kullanmaktadır. PAMAS meşgul mesajını RTS veya CTS kontrol paketlerinin uzunluğuna göre ayarlar. Bu arada başka bir RTS gelirse bunu kontrol kanalları ile halletmekte ve veriye bir şey olmasını önlemektedir [14]. Düğümün iletilecek paketi varsa RTS iletilir ve CTS bekleme moduna geçer. Buna rağmen eğer komşu paket alırsa bu komşu meşgul tonla (RTS/CTS'nin iki katı) karşılık verir. Bu ton CTS'nin alımıyla çakışacaktır. Bu düğümü BEB durumuna geçirecek ve düğüm paket iletemeyecektir. Eğer hiçbir komşu meşgul ton iletmezse ve CTS doğru şekilde varırsa, iletim başlar ve düğüm paket gönderme moduna geçer.

RTS ileten düğüm CTS mesajı almazsa, BEB durumuna geçer ve RTS'yi yeniden iletmeyi bekler. Buna rağmen eğer bazı komşular düğüme RTS gönderirse, BEB modu bırakılır, CTS iletilir ve paket bekleme moduna geçilir. Paket geldiğinde paket alma moduna geçer. Eğer beklenen zamanda bir şey duyulmazsa, boş moda geri döner. Bir düğüm paket aldığında, paket alma moduna geçer ve hemen meşgul ton iletir. Eğer düğüm diğer düğümlere doğru RTS iletimi duyarsa yâda paket aldığı periyod boyunca herhangi bir zaman kontrol kanalı üzerinde bir gürültü duyarsa, meşgul tonu iletir [15]. Bu sayede RTS ileten komşu beklenen CTS'yi alamayacağını bilir. Bu sayede komşu iletimi bloklanır.

PAMAS cihazları iki durumda gücü kapatırlar. Bunlar, ya gönderecek verisi yoktur ya da komşuları başka bir düğüm ile veri gönderimine başlamışlardır. Birinci durumda enerjinin korunması işlemi, cihazın her hangi bir bozulmadan dolayı veri mesajını alamaması durumunda düğümün kapatılması ile gerçekleştirilir. İkinci durumda ise cihaz kendinde veya komşusundan çarpışma sonucu almadan iletim veya alım yapmaz. Uyku süresinin hesaplanması için ise gelen başlangıç veri mesajında iletimin süresi de gönderilir ve bir cihaz istem dışı alım yaparak başlangıç verisinde bu süreyi alır ve uyku zamanını hesaplar. Bu protokolde her düğüm kendini kapatma işlemini bağımsızca yapar. Bir düğüm eğer komşusu iletim yapıyorsa bunu bilir çünkü o iletimi duyar. Benzer şekilde bir düğüm boş olmayan iletim kuyruğunu bilmektedir, çünkü bir ya da birkaç komşusu paket almaya başladığında meşgul ton iletmeye başlarlar. Bu sayede bir düğüm kolayca ne zaman kapanacağına karar verebilmektedir [14,15]. Neticede paket gecikmeleri düğüm kapanmalarının sonucu olarak artmamaktadır. Bu düğümün kapalı olduğu periyodun sonucu olarak artmaktadır. Bu protokol ayrıca gizli terminal problemini yakalamaktadır.

2.3.2. CC-MAC

CC-MAC(Spatial Correlation-Based Collaborative Medium Access Protocol-Mekânsal İlişki Tabanlı İşbirlikçi Ortam Erişim Protokolü) protokolünde

bütünlüğe dayalı MAC protokol tasarımı yapılmıştır. Bu protokol ortam erişimini yakın konumlu algılayıcı düğümler tarafından oluşturulan lüzumsuz iletimden korumaktadır. Mekânsal CC-MAC protokolü işbirlikçi algılayıcı düğüm iletimini regüle etmekte ve bozulma kısıtlaması bir olay hakkında bilgi gönderen düğümlerin sayısı azalsa bile başarılabilmektedir. Düğümlerin yerleri zekice seçilerek, bozulma azaltılabilmektedir. Bu amacı sağlamak için, INS algoritması baz istasyonunda yer alan, bozulma kısıtlaması için ilişki çapına karar vermektedir. Daha sonra bu bilgi ağ kurulumu esnasında her bir algılayıcı düğüme her yöne yayım ile gönderilmektedir. Her bir algılayıcı düğümde uygulanan CC-MAC protokolü dağıtık bir MAC yapısı kullanmaktadır [14,16]. CC-MAC KAA'da gereksiz bilgi akışını durdurmak için ilişki yarıçapını kullanmaktadır. Belirli bir kaynak düğüm kendi olay kaydını baz istasyonuna ilettiğinde, onun bütün ilişkili komşuları bozulma kısıtlamasına bağlı olarak gereksiz bilgiye sahip olmaktadırlar. Bu gereksiz bilgi, eğer gönderilirse KAA kaynaklarını harcamaya ek olarak ilişki bölgesindeki toplam gecikmeyi ve çekişmeyi arttırmaktadır.

Mekansal CC-MAC protokolü baz istasyonuna filtrelenmiş veriyi öncelikli olarak iletme ve böylece gereksiz bilgi iletimini önlemeye çalışmaktadır. KAA'da algılayıcı düğümlerinin veri üretme ve yönlendirme olmak üzere çift işlevselliği bulunmaktadır. Bundan dolayı ortam erişimi iki nedenden gerçekleşir. CC-MAC protokolü kaynak ve yönlendirme fonksiyonelliğine bağlı olarak iki bileşen içermektedir. E-MAC (Event MAC-Olay MAC) ilişkili kayıtları filtrelemekte ve N-MAC (Network MAC-Ağ MAC) yönlendirilen paketlerinin önceliklendirilmesini garanti etmektedir. Daha belirgin olarak bir düğüm E-MAC'i kendi algılama okumasını baz istasyonuna iletmek istediği zaman kullanırken, N-MAC'i paket alıp onu sıradaki hopa iletmeye çalışınca kullanmaktadır [16].

2.3.3. STEM

STEM(Sparse Topology and Energy Management-Seyrek Topoloji ve Enerji Yönetimi) karşılaşma tabanlı protokollerdendir. STEM kapasite korunumu için çalışmaz, böylece büyük enerji korunumu sağlar ve yönlendirme bundan etkilenmez. STEM düşük güç tüketimli bir radyo yerine düşük görev çevrime sahip bir radyoya sahiptir. Bu protokolde en önemli olay algılayıcı düğümlerin bağımsız bir şekilde uyuma moduna geçebilmeleridir. STEM protokolünde enerji korunumunu, uyku modunda olan bir komşu düğüme kısa bir mesajla ve düşük güç harcayarak uyanmasını sağlamak şeklinde gerçekleştirilmektedir. Bir düğüm komşunu iki şekilde uyandırabilir bunlar bir işaret sinyali mesajı gönderilmesi (STEM-B) veya uyanma tonu ile mümkündür (STEM-T) [17].

İletişime geçmek isteyen başlatıcı düğüm, hedef düğüm olarak adlandırılan düğümü uyandırmaya çalışmak için o düğümün id'siyle bir sinyal yollar. Gerçekte bu başlangıç düğümü ve hedef düğümü arasında bir bağlantı aktive etmeye çalışmaktadır. Hedef düğüm sinyal alır almaz başlangıç düğümüne karşılık verir ve her ikisi bu noktada radyolarını açık tutar. Eğer paket daha sonra gönderilmek istenirse hedef düğüm sıradaki hop için başlatıcı düğüm olacak ve işlem böylece devam edecektir [14,17]. Her iki düğümde radyolarını açmak için bir bağlantı kurduklarında bağlantı aktiftir ve sonraki paketler için kullanılabilmektedir. Güncel veri iletiminin uyandırma protokolü ile karışmaması için STEM bunları farklı frekans bandında, her bir bantta farklı bir radyo kullanarak göndermektedir. Hiyerarşik olasılıklı kümeleme yaklaşımları için, STEM küme başının enerjisini azaltmak için kullanılabilmektedir. STEM kapasite korunumu için çalışmamaktadır. Böylece büyük enerji korunumu sağlamakta ve yönlendirmeyi etkilememektedir.

STEM ile ilgili olarak bir yaklaşım ise ana radyolarını açmış olan düğümleri uyandırmak için ayrı sayfalama kanalının kullanılmasıdır. Buna rağmen, sayfalama kanalı radyosu bazı nedenlerden dolayı uyku moduna konulamamaktadır. Bu yaklaşımda düzenli veri iletişimi için kullanılanlara göre sayfalama kanalı daha

düşük güçtedir. STEM sayfalama kanalını düşük güç tüketimli bir radyo yerine, düşük görev çevrim radyolu bir radyoya sahip olarak canlandırmaktadır.

2.3.4. TICER ve RICER

TICER(Transmitter Initiated Cycled Receiver- Çevrimsel Alıcıyı Başlatan İletici) protokolü STEM-B durumuna benzer çalışmaktadır. Bir algılayıcı düğüm iletim için veri paketine sahip değilse, kanalı izlemek için T periyoduyla uyanır ve uyanma süresi tonundan sonra tekrardan uyku moduna döner. Eğer düğümün, protokol yığınının üst katmanında oluşturulan veya başka bir düğüm tarafından yönlendirilen bir gönderecek paketi olursa uyanır ve ton süresi kadar kanal izlemesi yapar. Eğer kanalda devam eden bir iletim duymazsa hedef düğüme RTS sinyali göndermeye başlar ve her RTS iletiminden sonra cevap için TI zamanı kadar kanalı izler. Kendi uyanma planına göre uyanan hedef düğüm, hemen RTS alır ve CTS ile kaynak düğüme karşılık verir. CTS sinyali alındığında kaynak düğüm veri paketini iletmeye başlar. Veri paketinin doğru bir şekilde alınmasından sonra hedef düğüm kaynak düğüme oturumun bitiğini bir ACK sinyali göndererek haber verir [14,18]. TICER protokolü tasarımı sırasında dikkate alınan bazı konular şu şekildedir;

- Uzun tonun kullanımı, performans artırımı olmadan daha fazla izleme gücü harcanmasına sebep olmaktadır. Tonun güç korunumu için olabildiğince kısa olması gerekmektedir. Bir RTS almak için ihtiyaç duyulan minimum zamanın alt sınır olarak ayarlanması gerekmektedir. Benzer şekilde CTS alımı için ihtiyaç duyulan minimum zaman TI için ayarlanmalıdır.
- İletim gücünü minimize etmek için, kontrol paketleri RTS, CTS, ACK olabildiğince kısa yapılmalıdır. Bu paketler kaynak ve hedef düğümlerin yerel MAC adreslerini ve edinimler için öntakı içermektedirler.
- İletilen gücü korumak için RTS iletiminde kullanılan frekans olabildiğince düşük olarak tutulmalıdır. Fakat periyodu tondan büyük olmamalıdır, eğer olursa hedef düğüm 2 RTS arasında uyanıp tekrardan uyuyabilir ve randevuyu kaçırabilir. Bu nedenle periyot tondan daha kısa ayarlanmalıdır.

RICER (Receiver Initiated Cycled Receiver-Çevrimsel Alıcıyı Başlatan Alıcı) TICER planına benzer şekilde bir algılayıcı düğümün iletecek paketi yoksa T periyodunda uyanır. Sonra uyanık olduğunu bildirmek için uzunluğu Tb olan kısa uyanma işaret sinyali yollar ve TI süresince cevap için kanalı izler. Eğer cevap yoksa düğüm tekrar uyku moduna geçer. İletecek veriye sahip olan kaynak düğüm uyanık kalır ve hedef düğümden bir uyanma sinyali bekleyerek kanalı izler. Cevabı aldığında veri paketini iletmeye başlar. Oturum veri paketini doğru bir şekilde aldıktan sonra hedef düğümün kaynak düğüme ACK yollamasıyla bitirilir. RICER protokolünün tasarım kuralları TICER protokolüyle benzerdir[18].

- TI, hedef düğümün veri paketinin doğru bir şekilde geldiğini anlaması için yeteri kadar uzun olmalıdır.
- Uyanma işaret sinyali uzunluğu olabildiğince kısa olmalıdır.

2.3.5. B-MAC

B-MAC(Berkeley Medium Access Control Protocol- Berkeley Ortam Erişim Kontrol Protoklü) STEM-T durumuna benzemektedir ve uyuyan komşularını uyandırmak için bir ton kullanmaktadır. Bu protokol CSMA tabanlı bir protokolüdür. B-MAC düşük güç işlevini sağlamak için, uyarlanabilir öntakı örnekleme planı kullanmaktadır. Bu sayede görev çevrimini ve boşta dinlemeyi azaltmaktadır. B-MAC çalışma esnasında tekrar konfigürasyonu desteklemekte ve performansı optimize etmek için çift yönlü arayüz sağlamaktadır.

B-MAC protokolü mesaj transferi için çok uzun öntakı kullanmaktadır [14,19]. Öntakının büyük olması ve transferi ile enerji korunumu, gecikme açısından optimum değiş-tokuş ortaya çıkmaktadır. B-MAC CCA(Clear Channel Assesment), kanal çözümlemesi için paket geri çekilmesi, güvenlik için bağlantı katmanı alındı bilgilendirmesi, düşük güç iletişimi için düşük güç dinleme planını kullanmaktadır. B-MAC bir arayüz kümesine sahiptir. Bu arayüzler standart mesaj arayüzüne ek olarak operasyonları ayarlama imkânı vermektedir. Bu

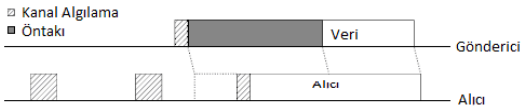
arayüzler CCA, ACK, geri çekilme ve LPL gibi B-MAC mekanizmasının içerdiği durumlar için ağ servislerine ayarlama izni vermektedir.

Çevreye bağlı olarak değişen gürültü ortamı için B-MAC yazılımsal otomatik kazanım kontrolünü uygulamaktadır. Sinyal güç örnekleri kanal boş olarak varsayıldığında alınmaktadır. Kanalın boş olması ise bir paket iletildikten hemen sonra radyo yığınının veri yolunda geçerli veri almadığı zaman olmaktadır. Örnekler daha sonra FIFO kuyruğuna girmektedir. Kanal örnekleme periyodu sırasında aykırılık olursa, B-MAC kanalı temiz olarak deklare eder. Çünkü geçerli paket hiçbir zaman gürültü tabanından önemli derecede küçük aykırı bir değere sahip olamaz. Eğer 5 örnek alınır ve aykırı bir değer bulunmaz ise kanal meşguldür [19]. Eğer CCA etkinse, B-MAC paket gönderirken başlangıç kanal gecikmesini kullanmaktadır.

Ayrıca B-MAC protokolü algılayıcı düğümlere protokol üzerindeki birçok işlemsel değişkenleri değiştirme imkânı sunmaktadır. Örneğin gecikme ve geri çekilme zamanı gibi. B-MAC geri çekilme zamanını ayarlamaz bunun yerine paket gönderen servise MacBackoff arayüzüyle bir olay sinyallenir. Servis bir başlangıç geri çekilme zamanı döner yâda sinyallenen olayı yoksayar. Eğer yok sayarsa, küçük rastgele geri çekilme kullanılmaktadır. Başlangıç geri çekilmesinden sonra, CCA aykırı değer algoritması çalışmaktadır. Kanal temiz değilse, bir olay tıkanıklık geri çekilme zamanı için servisi sinyaller. Eğer geri çekilme zamanı verilmezse, küçük rastgele geri çekilme kullanılmaktadır.

B-MAC'te kullanılan teknik Alohadaki öntakı örneklemeye benzer fakat farklı radyo karakteristiğine uygun hale getirilmiştir. Düğüm uyandığı zaman, radyoyu açar ve aktivite kontrolü yapar. Eğer aktivite algılanırsa, düğüm gücünü açar ve gelen paketleri almak için gerekli olan zaman kadar uyanık kalır. Aldıktan sonra, düğüm tekrar uykuya döner. Eğer paket alınmaz ise bir zaman aşımı işlemi düğümü tekrar uykuya geçirir. B-MAC protokolünde enerji kullanımını azaltmak için sinyal kuvvetinin kullanımına yönelik bir eşik değeri belirlenir. Güvenli bir

şekilde veri almak için, başlangıç öntakısı uzunluğu kanalda aktivite olup olmadığını anlama aralığına eşlenir. Boş dinleme, düğüm kanalı örneklemek için uyandığında ve aktivite olmadığı zaman gerçekleşir. Ayrıca algılayıcı düğümler iletim gerçekleştirmeden önce rastgele bir süre bekleyerek senkronizasyonu korurlar ve o andaki iletişim ile kendi gönderdikleri paketlerin çakışmasını önemli ölçüde önlerler. Geleneksel kablosuz ağlarda bulunan gizli terminal problemi bu protokolünde problemlerinden bir tanesidir. Planlı protokollerin aksine B-MAC protokolü kanala erişim için gecikme mesajına sahip değildir. Şekil 2.1'de B-MAC protokolünün iletim şekli gösterilmektedir [14,19].



Şekil 2. 1. B-MAC veri transferi

2.3.6. Wise-MAC

Wise-MAC(Wise Medium Access Control-Akıllı Ortam Erişim Kontrolü) STEM ve bu protokollere ilgisi olan protokollere benzer protokoldür. B-MAC ile aynı zamanda geliştirilmiş ve geliştirilirken benzer teknikler kullanılmıştır. Fakat enerji tüketimini azaltmak için algılayıcı düğümlerin komşularının örneklemelerinin hatırlanması olayını benimsemiştir. Burada var olan ekstra bir alan ise ACK paketleri içindedir ve bu alan ile algılayıcı düğümler komşu düğümlerine bir sonraki kanal örneklemelerinin zamanını gönderirler. Bir düğüm komşusunun bu örnekleme zamanını öğrenerek alıcı taraf uyanana kadar öntakı iletimini geciktirebilir. Komşuların örneklemeleri için kullanılacak bir ek bellek ve ACK içinde kullanılacak ek bir alan ile algılayıcı düğümler öntakı gönderim süresini azaltabilirler ve bu sayede istem dışı alımlarda azaltılabilmektedir [14,20]. Wise-MAC' te kullanılan yöntemlerden biri olan TDMA veri kanalına erişim için kullanılmakta, bir diğeri olan CSMA kontrol kanalına erişim için kullanılmaktadır. Wise-MAC ağda düşük trafik koşullarında düşük güç tüketimi, yüksek trafik koşullarında da yüksek enerji verimliliği sunmaktadır.

2.3.7. CSMA-MPS

CSMA-MPS(CSMA With Minimal Preamble Sampling- CSMA ile Minimum Öntakı Örneklemesi) protokolünde B-MAC ve Wise-MAC protokollerinde bulunan gecikme ve enerji konusu geliştirilmiştir. CSMA-MPS yaklaşımı çok verimli uyanma mekanizması kullanmakta ve bu sayede düşük enerji tüketimini optimize eden bir MAC protokolüdür. CSMA-MPS, STEM ve WiseMAC protokollerinin uyanma mekanizmalarının verimliliğini arttırmak için geliştirilmiş ve bu işlem her iki protokolün en iyi özelliklerinin birleştirilmesiyle sağlanmaya çalışılmıştır. CSMA-MPS protokolü STEM protokolünün alternatif iletim ve alımını kullanmakta ve bunları WiseMAC protokolünün daha verimli uyanma özelliğiyle birleştirmektedir. Alternatif iletim ve alım kullanmak en kötü durum uzunluğu ile bir öntakı gönderme ihtiyacı olmaksızın neredeyse anında senkronizasyona izin vermektedir [14,21]. STEM-B ve TICER protokollerine benzer şekilde uzun öntakı göndermek yerine kaynak düğümler küçük kontrol mesajları göndermekte ve alıcıdan bir cevap var mı diye hattı dinlemektedirler. CSMA-MPS protokolünde, ağda tüm düğümler ortamı kanalda aktivite olup olmadığını kontrol etmek için kısa sabit periyot sürecince örneklemektedir. Komşularından birisiyle iletişime geçmek isteyen bir düğüm, komşularının örnekleme planına sahip olabilir veya olamayabilir. Eğer sahip değilse hemen CSMA algoritması başlatılır, diğer durumdaysa eğer bir saat tahmini varsa alıcı kendisinin periyodik dinleme aralığına başlamadan hemen önce taşıyıcının doğru zamanda algılamaya başlaması için bir kaç zaman geriye gönderilmektedir. Örnekleme planı ve son senkronizasyonun zamanı bilindiğinde, kaynak örnekleme planını bekler ve minimum boyutlu uyanma dizisiyle ile aynı zamanda uyanma veri mesajı dizisi gönderir.

Bu protokolde ağ genelinde senkronizasyon ve periyodik yerel senkronizasyon mesajı gerekmez. Sadece mesajlar değiş tokuş edildiğinde, uyanma planı hem veri çerçevesinde hem de ACK çerçevesinde bulunmaktadır. CSMA-MPS protokolünde kaynak düğüme komşusunun örneklemesini belirleme imkânı

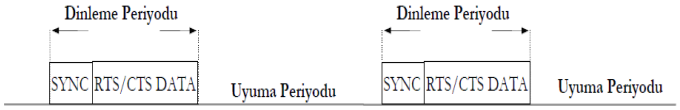
verilmekte ve ACK içinde ek bir alana ihtiyaç duyulmamaktadır. Ayrıca küçük kontrol mesajları RTS ve CTS kontrol mesajları işlevini yerine getirmektedir [21].

2.3.8. S-MAC

S-MAC(Sensor Medium Access Protocol-Algılayıcı Ortam Erişim Protokolü) protokolü CSMA tabanlı bir protokolüdür. Algılayıcı düğümler anlık ağlara benzemezler, en belirgin özellikleri ise uzun bir süre uyusalar bile bir aktivite belirlendiğinde hemen uyanma moduna geçmeleridir. S-MAC enerji tüketimini azaltmak ve kendi kendine yapılandırma olayını desteklemek için 3 tane yeni teknik kullanmaktadır. Ortamın gereksiz yere dinlenmesi sırasında harcanan enerjiyi azaltmak için düğümler periyodik olarak uyku moduna geçirilmektedir. Komşu düğümler sanal kümeleme içinde olup uyku planları otomatik olarak senkronize olmaktadır [22]. PAMAS protokolünden esinlenerek S-MAC protokolünde bir düğüm diğer düğümler ile iletişim yaparken kendi radyosunu kapatabilmektedir. PAMAS protokolünden bir farklı yönü kanal içi sinyalleşme tekniğini kullanmasıdır. Ayrıca S-MAC protokolü algılayıcı ağ uygulamaları için çakışmadan kaynaklanacak gecikmeyi azaltmak için mesaj aktarma metodunu kullanmaktadır. S-MAC protokolünün birinci amacı enerji tüketimini azaltmak iken bunun yanında iyi bir ölçeklenebilirlik ve çakışmadan kaçınma yeteneğine de sahip olduğu görülmektedir. İyi ölçeklenebilirlik ve çakışmadan kaçınma için birleşik planlamadan ve rekabet planından yararlanılmıştır [14,22].

S-MAC protokolünde kullanılan mesaj aktarma kavramı ile büyük verilerin iletilmesi sağlanmaktadır. Bu sayede en temel bilgi olan büyük paketlerin küçük parçalara ayrılarak gönderilmesi durumu ortaya çıkmaktadır. Buradaki en büyük problem ise bir düğüm büyük boyutta veri gönderiyorsa ortama diğerlerinden daha fazla erişecektir bu da her açıdan adaletsizlik ortaya çıkarmaktadır. S-MAC protokolünde bu problem mesaj aktarma kavramı ile çözülmeye çalışılmıştır [22].

Gecikme ise düğümde hangi uygulamanın çalıştığına göre ya önemlidir ya değildir. Ağda her hangi bir akış yok ise çok düşük bir veri iletişimi ortaya çıkacaktır. Bu yüzden çoğu zaman düğümler boş durumundadırlar. Bu durumda çok az gecikmeler fazla önemli olmamakta biraz gecikme olsa bile bu olay enerji korunumu için kullanılabilmektedir. Dolayısıyla S-MAC protokolünde düğümler ortamın gereksiz yere dinlenmesi durumunda iken düğümleri periyodik olarak uyku moduna göndermektedir. Dolayısıyla dinleme uyuma zamanları sabit ve periyodiktir. Bu sayede ortamın gereksiz yere dinlenmesinden ortaya çıkacak olan enerji tüketimi S-MAC protokolü sayesinde azaltılmaktadır [22]. Ancak bu durumda gecikme artmaktadır. Kısaca gönderen düğüm veri göndermek için alıcı düğümün uyanmasını beklemektedir. Düğümlerin işbirliği yapabilmesi ve beraber hareket edebilmesi için çok iyi bir senkronizasyona ihtiyaç duyulmaktadır. S-MAC'e ait zamanlama diyagramı Şekil 2.2'de gösterilmiştir.



Şekil 2.2. S-MAC mimarisi

S-MAC protokolünde periyodik dinleme ve uyuma işlemi için bir zamanlayıcı kullanılmıştır. Düğümler uyku moduna geçince zamanlayıcı çalışmaya başlıyor ve belirlenen süre dolunca düğümler uyanıyorlar. Bütün düğümler kendi dinleme ve uyuma zamanlarını belirleyebilmektedirler. Bu önerilen yöntemde komşular arasındaki saat sapmalarına çözüm getirmek için senkronizasyon mutlaka gereklidir. Bu sayede komşular kendi aralarında beraber uyku moduna geçip beraber uyanmaktadırlar. Bir iletişim başladıktan sonra uyku zamanı gelse bile iletişim bitmeden uyku moduna geçilmemektedir [14,22]. 802.11 gibi RTS/CTS kontrol paketlerini kullanmaktadırlar. Senkronizasyon için sanal kümeleme kullanılmaktadır. Periyodik dinleme/uyuma moduna geçmeden her düğüm bir plan seçmeli ve bunu komşularına bildirmelidir. Her düğümün bir plan tablosu bulunmakta ve bilinen tüm komşuların planları bu tabloda güncellenmektedir.

2.3.9. T-MAC

T-MAC(Timeout Medium Access Control- Zaman Aşımı Ortam Erişim Protokolü) protokolü CSMA tabanlı bir protokolüdür. S-MAC protokolünden esinlenmiştir. T-MAC boşta dinlemeyi KAA için minimize etmektedir. Sabit uzunluktaki görev çevrimi yerine bir zamanlayıcı koyarak aktif periyodun sonlanıp sonlanmayacağına karar vermektedir. Farklı yükteki en uygun aktif zamanı korumak için dinamik olarak onun uzunluğuna karar verilmektedir. Aktif zaman sezgisel yolla yani hiçbir şey işitilmediğinde sonlandırılmaktadır. Her düğüm komşularıyla iletişime geçmek için periyodik olarak uyanır ve sonra bir sonraki çerçeveye kadar tekrardan uykuya geçer. Bu sırada yeni mesajlar kuyruğa alınır. Düğümler birbirleriyle iletişimi RTS, CTS, ACK paketlerini kullanarak gerçekleştirirler. Aktif periyotta olduğu sürece bir düğüm dinlemeye ve potansiyel iletime devam edecektir. Aktif periyot, TA zamanı içerisinde bir aktivasyon olayı oluşmazsa sona erecektir [14,23]. Aktivasyon olayı ise şu şekilde oluşur;

- Periyodik çerçeve zamanlayıcısının tetiklenmesi
- Radyodan herhangi bir verinin alınması
- Radyo üzerinde iletişimin algılanması
- Düğümün kendi verisinin veya ACK iletiminin bitmesi
- Komşularının veri alış verişinin sonlandığına dair RTS ve CTS paketlerinin önceden dinlenmesiyle elde edilen bilgi

Düğüm eğer aktif periyotta değilse uyuyacaktır. Sonuç olarak TA her çerçeve başına minimum boşta dinleme miktarına karar vermektedir. Çerçeve senkronizasyonu S-MAC protokolünde açıklanan sanal kümelemeden esinlenmiştir. Bir düğüm ne zaman uyansa beklemeye ve dinlemeye başlar. Eğer zamanın belirli bir diliminde bir şey işitmezse, çerçeve planını seçer ve bir sonraki çerçevenin başlangıcını içeren bir SYNC paket gönderir. Eğer bir düğüm başlama sırasında başka bir düğümden bir SYNC işitirse, bu SYNC paketindeki planı takip eder ve benzer şekilde kendi SYNC paketini gönderir. Düğümler kendi SYNC'lerini arada bir tekrar gönderirler [23].

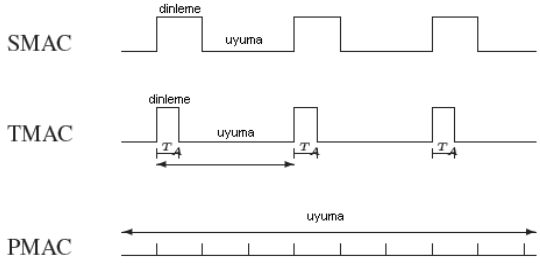
Düğümler veri iletimine sadece kendi aktif zamanlarının başlangıcında başlamalılar. O zaman aynı plana sahip olan komşular ve ekstra olarak bir planı benimseyen komşular uyanırlar. Bu plan her yöne yayınların sadece bir kere iletilmesine olanak tanımaktadır. T-MAC mimarisi Şekil 2.3'te gösterilmiştir.

Şekil 2.3. T-MAC mimarisi

Sanal kümeleme tekniğinin uygulanması kolaydır. Ancak sanal kümelemede bulunan dinleme periyotları senkronizasyonunu bozmaktadır. Bu problemden dolayı T-MAC protokolünde erken uyuma problemi ortaya çıkmaktadır.

2.3.10. P-MAC

P-MAC(Pattern Medium Access Control- Modelli Ortam Erişim Kontrolü) protokolü CSMA tabanlı bir protokolüdür. P-MAC protokolünde sabit uyuma uyanma yerine algılayıcı düğümlerin uyanıp uyumasına uyarlamalı olarak karar verilmektedir. Düğümlerin kendi trafiğine ve komşularının trafiğine bakılarak planlara karar verilmektedir. P-MAC, S-MAC protokolü gibi zaman dilimli bir protokoldür. P-MAC protokolünde bir düğüm bir zaman dilimi süresince uyku veya uyanık durumda olabilir [14,24]. Şekil 2.4'te S-MAC T-MAC ve P-MAC protokollerinin algılayıcı ağda trafik olmadığı durumda boşta dinleme periyodunun uzunluğunu göstermektedir.



Şekil 2.4. S-MAC, T-MAC, P-MAC karşılaştırması

PMAC protokolünde bir algılayıcı düğüm komşularındaki aktivite hakkındaki bilgileri, desenlerini elde etmeden önce alır. Bu desenlere bağlı olarak ağda trafik olmadığı zaman bir algılayıcı düğüm birkaç zaman dilimi için kendini uzun bir uykuya koyar. Komşusunda bir aktivite olduğu zaman bunu bu desenler aracılığıyla bilir ve gerektiğinde uyanır. Uyuma-uyanma deseni bitlerin string olarak gösterilmesi şeklindedir ve birçok zaman diliminde algılayıcı düğüm için geçici uyku–uyanma planını gösterir. String'te bulunan 1 değeri bir zaman dilimi süresince düğümün uyanık duracağını gösterir, 0 ise düğümün uyku durumunda olacağını gösterir. Örneğin bir düğüm için 001 deseni geçici planında 2 ardışık zaman diliminde uyku durumunda olacağını ve 3. dilimde ise uyanık olacağını göstermektedir. Dolayısıyla desenler düğümün uyku ve uyanma zamanlarına ve protokolün performansına etki ederler. Bir düğümün deseni onun uyku ve uyanma zamanını değiştirmektedir [24].

Pj, j düğüm deseninin ikili string gösterimi olsun. Bu desen düğüm j ile N zaman dilimi üzerinden bağlantılıdır. N zaman dilimlerinin sırasına periyot denmektedir. Eğer Pj'nin uzunluğu N'den küçükse, kalan süresi boyunca desen tekrarlanır. Örneğin Pj=01 ve N=5 ise j düğümünün sonraki 5 zaman dilimi için geçici planı 01010 şeklinde olur ve düğüm dilim 1,3 ve 5'te uyku durumunda, kalan 2 ve 4'üncü dilimlerde ise uyanık durumda olur. PMAC protokolünde desen 0^m1 olarak sınırlandırılmıştır ve m=0,1…N-1 şeklindedir. Desendeki 0 bitlerinin sayısı m ile gösterilir, desene sahip olan düğüm etrafındaki trafik yoğunluğunu belirtir. Büyük m trafik yoğunluğunun az olduğunu, küçük m ise trafik yoğunluğunun fazla olduğunu gösterir. Trafik koşullarına adaptasyon için, bir düğümün deseni düğümdeki mevcut yerel trafik bilgilerini kullanarak her periyotta güncellenir ve her periyodun sonunda değiştirilir. Eğer düğüm j zaman dilimindeki desen bitine bağlı olmaksızın herhangi bir zaman diliminde iletecek bir veriye sahipse sonraki sıralamada desen 1'e geri çekilir. Bu j düğümüne trafik yüküyle baş etmek için düğümü hızlıca uyandırma imkânı tanır. Sonraki güncelleme bu yeni desenle başlamaktadır. Sonraki periyot için oluşturulan yeni desenler, o anki periyodun sonuna kadar düğümler tarafından her yöne yayınlanır [14,24].

Bu desen değiş tokuşunun sağlanabilmesi için, zaman STF(Super Time Frames-Süper Zaman Çerçeveleri)'ye bölünmüştür. Her STF, düğümün geçici uyku uyanma planını içermektedir. Her STF 2 alt çerçeveye sahiptir. Birincisi PRTF(Pattern Repeat Time Frame-Desen Tekrarlama Zaman Çerçevesi) olarak adlandırılır ve her düğüm kendi o anki planını tekrarlıyorsa kullanılmaktadır. Bütün N zaman dilimlerinin sonunda, PRTF bütün algılayıcı düğümlerin uyanık olduğu süre zarfında bir tane ek zaman dilimine daha sahiptir. STF'nin ikinci alt çerçevesi PETF(Pattern Exchange Time Frame- Desen Değiş Tokuş Zaman Çerçevesi) olarak adlandırılır. Buda komşu düğümler arasında yeni desenlerin değiş tokuşu yapıldığında kullanılmaktadır. PETF'de TE süresince birçok zaman dilimine bölünmektedir. Belirli bir PRTF süresince en son oluşturulan desen, sonraki PRTF'nin deseni haline gelir ve bu komşulara PETF süresince yayınlanır. Eğer j düğümü PETF süresi esnasında komşularından herhangi birinden yeni desen almazsa, onların eski planlarını tekrarlar. Zaman dilimini kapsayan TR bütün veri iletimini gerçekleştirme için yeteri kadar uzun seçilmelidir(Çekişme Penceresi + RTS + CTS + DATA + ACK). N için seçenek, PRTF'nin zaman dilimi sayısını uygulamaya bağlıdır. Eğer N yüksekse, algılayıcı düğümler için daha fazla uyuma zamanı mümkündür ve böylece daha fazla enerji korunumu sağlanabilmektedir. Fakat bu aynı zamanda veri iletimindeki gecikmeyi de arttırabilmektedir. PETF'deki zaman dilimleri sayısı bir algılayıcı düğümün sahip olabileceği maksimum komşu sayısına göre ayarlanır [24]. PETF'deki zaman dilimi TE'yi kapsayan desenin her noktaya yayını sağlamak için yeteri kadar uzun seçilmelidir.

2.3.11. DSMAC

DSMAC(Medium Access Control With A Dynamic Duty Cycle For Sensor Networks-Algılayıcı Ağlar İçin Dinamik Görev Çevrimli Ortam Erişim Kontrolü) protokolü S-MAC protokolünden esinlenmiştir. DSMAC, iki performans metriği arasında fazla ek yüke maruz kalmadan iyi bir değiş tokuş sağlamaktadır. Ayrıca DSMAC uygulama gereksinimlerinin önceki bilgilerine ihtiyaç duymadan değişen trafik koşullarındaki görev çevrimlerini ayarlayabilmektedir. Daha belirgin olarak,

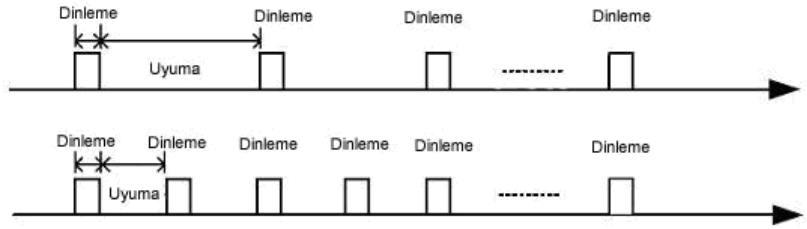
uygulamaya bağlı olarak, bir algılayıcı öncelikle bir tarama fazı uygular, algılayıcı belirli bir periyot kanalı dinler, eğer varsa bir SYNC paketi tarafından içine alınmış var olan uyuma-uyanma planını benimsemeye çalışır. Fakat eğer periyodun sonuna kadar bir SYNC paketi alınmaz ise, algılayıcı düğüm kendinin komşuluktaki ilk aktif algılayıcı olduğunu varsayar ve özgürce planı seçer. Sonra düğüm mevcut düğümleri, kendi planını SYNC paketlerinin her yöne periyodik yayın yapmasıyla bilgilendirir. Her düğüm komşu düğümleri için bir senkronizasyon tablosu tutar. Uyuma uyanma planına karar verildiği zaman, her düğüm saat senkronizasyon bilgilerini içeren SYNC paketlerini her yöne yaymaya başlar [14,25]. Ne zaman bir düğüm bir SYNC paketi duyarsa, kendi senkronizasyon tablosunu günceller ve kendi zamanlayıcısını SYNC paket oluşturucusuna göre ayarlar.

DSMAC protokolü uyuma uyanma çevrim zamanını algılayıcının o anki enerji kullanım verimliliği ve ortalama gecikme deneyimine göre dinamik olarak ayarlamaya çalışmaktadır. Bir hop gecikme, paketin kuyruğa alımı ve başarılı şekilde gönderimi arasındaki fark olarak tanımlanmaktadır. Alıcı düğümün ortalama gecikmesi o anki SYNC periyodunda toplanan bütün bir hop gecikme değerlerinin ortalama değeridir [25]. Bu ortalama gecikme değeri alıcı düğümler için o anki trafik koşullarının yaklaşık tahmininin yapılmasını ve parametreleri gösterme hizmetini yapma imkânı vermektedir.

DSMAC protokolü çift evreli ayarlama modülü kullanmaktadır. Bunun dışında her alıcı düğüm ayrıca kendi enerji tüketim verimliliği ve ortalama gecikmenin kaydını tutmaktadır. Her bir paket iletimindeki enerji tüketim miktarının izlenmesiyle enerji tüketimi seviyesi ölçülebilmektedir. Sabit görev çevrimi kullanmayan DSMAC'te bütün algılayıcılar başlangıçta ortak temel hizmet görev çevrimini benimsemektedirler.

Alıcı algılayıcı düğüm için eğer gecikme kabul edilemez durumdaysa, dinleme zaman periyodunu değiştirmeden uyku zaman periyodunun uzunluğunu kısaltarak orijinal görev çevrimini ikiye katlama kararını tek taraflı olarak yapar. Böylece,

kendi görev çevrimini arttıran düğümün diğer göndericilerden gelen paketleri uyku durumunda olup bloklama yerine alma şansı daha fazla olmaktadır [25]. Şekil 2.5'te DSMAC mimarisi gösterilmektedir.



Şekil 2.5. DSMAC mimarisi

Ayrıca SYNC paketinde, SYNC görev çevrimi başlatıcısını gösteren yeni bir alan yer almaktadır. Düğüm SYNC paket yoluyla senkronizasyon bilgilerine ek olarak o anki görev çevrimini de her yöne yayınlar. Komşu algılayıcı düğümlerden SYNC paketini duyanlar kendi kuyruklarını kontrol ederler ve eğer gönderecek bir pakete sahiplerse ve SYNC paketindeki belirtilen görev çevrimi var olan plandan daha büyükse, kendi görev çevrimini ikiye katlarlar. Aksi takdirde eğer yeni bir plansa alıcı kendi plan tablosunu günceller. TE değeri enerji tüketim seviyesindeki üst sınırı gösteren bir eşiği temsil eder. Sadece geçerli güç tüketim seviyesi TE değerinin altındaysa, iki kat görev çevrimine izin verilir. Artan görev çevrimi daima birçok temel görev çevriminin üssel artanından oluşmaktadır. Bu nedenle eski plandaki orijinal dinleme periyodu yeni planda da aynı şekilde kullanılmaktadır [25]. DSMAC protokolünün sunduğu ek yük ise SYNC paketlerinde var olan ekstra görev çevrimi alanı ve ihmal edilebilen veri paketleri için gecikme alanı içermesidir.

2.3.12. TRAMA

TRAMA(Traffic-Adaptive Medium Access-Uyarlamalı Trafik Ortam Erişimi) KAA'lar için enerji verimli çakışma olmayan kanal erişimi için geliştirilen bir protokoldür. TDMA tabanlı bir protokolüdür. TRAMA enerji tüketimini tek noktaya yayın, çok noktaya yayın ve her yöne yayın iletiminde çarpışma olmamasını ve

düğümlere alma veya iletme yapmadıkları zaman düşük güç moduna geçmelerine izin vererek azaltmaktadır. TRAMA zamanı dilimli olarak kullanır ve belirli zaman diliminde hangi düğümün iletim yapacağına karar verme için her düğümdeki trafik hakkındaki bilgiler üzerinde dağıtık seçim planını kullanır. Planlı zaman dilimi kullanarak büyük veri mesajları için çekişmeyi önlemiş küçük ve periyodik kontrol mesajları için zaman dilimlerine rastgele erişimi desteklemiştir [14,26]. Düğümler TRAMA değiş tokuşunu kendilerinin iki-hop komşuluk bilgilerini ve iletim programlarını kronolojik sırayla kendi trafiklerindeki istenilen alıcı düğümleri belirlemek ve sonra her zaman dilimi sırasında iletecek ve alacak düğümleri seçmek için kullanırlar.

TRAMA topoloji bilgilerini paylaşmak için Komşu Protokolü(NP) ve düğümlerin kuyruklarında ne kadar trafiğe sahip olduğu bilgisini paylaşan ve aynı zamanda düğümlere iki-hop komşuluk bilgilerinin ve planlarının değiş tokuşuna izin veren plan değiş tokuşu protokolü(SEP) ve komşuluk ve plan bilgilerini kullanarak o anki zaman dilimini kullanacak gönderici ve alıcıları seçmek için ve diğer bütün düğümleri düşük güç modunda bırakma işlemi için uyarlamalı seçim algoritmasını(AEA) kullanır [26]. TRAMA veri ve sinyalleşme iletimi için tek zaman dilimli kanal kullanmaktadır. NP bütün düğümler boyunca tutarlı iki hop topoloji bilgilerini elde etmek için sinyal dilimlerini kullanarak rastgele erişim periyodu sırasında bir hop komşuluk bilgilerini komşu düğümler arasında yaymaktadır. Rastgele erişim periyodu sırasında, düğümler çekişmeli kanal edinimi işlemini yerine getirirler. Çerçevenin başlangıcıyla birlikte rastgele erişim kontrolü için kullanılacak zaman dilimleri oluşturulmakta ve planlı veri zaman dilimleri ise çerçevenin sonunda oluşturulmaktadır. İletim dilimleri çakışma olmadan veri değiş tokuşu ve ayrıca plan yayımı için kullanılırlar. Düğümler komşularıyla trafik tabanlı bilgileri veya planların değiş tokuşu için SEP protokolünü kullanmaktadırlar. Bir düğüm gerçek iletime başlamadan önce SEP protokolünü kullanarak planını duyurması gerekmektedir. SEP protokolü komşuların tutarlı plan bilgilerini tutar ve planları periyodik olarak güncelleme işlemini yerine getirir. Algılayıcı düğümler plan paketlerini sahip oldukları her çerçevenin son zaman diliminde iletirler ve bunun

içinde algılayıcı düğümün sonraki çerçevesinde kaç zaman dilimi olduğu ve ilgili bilgilerde bulunmaktadır [26]. AEA protokolü NP ve SEP protokollerinden elde ettiği bilgileri kullanarak çakışma olamadan iletişimde başarılı olmak için uygun ileticileri ve alıcıları seçmektedir. Çakışma olamadan gerçekleştirilecek iletimde enerji verimliliğinde başarılı olmak için belirli zaman dilimi için iletici ve alıcının seçilmesi bir gerekliliktir. AEA trafik bilgilerini kanalın verimli kullanımını arttırmak için kullanmaktadır.

TRAMA rastgele erişim moduna, her düğüm rastgele bir dilim seçerek iletimini yaparsa başlamaktadır. Düğümler ağa yalnızca rastgele erişim periyodunda girebilmektedir. Daha dinamik ağlarda rastgele erişim periyodu daha sık oluşmaktadır. Daha statik senaryolarda rastgele erişim periyotları arasında daha fazla zaman geçmektedir. Çünkü topoloji değişikliklerinin düzenlenmesi ihtiyacı arada bir gerçekleştirilmektedir. Bu tip algılayıcı ağlarda uygulamanın tipine bağlı olarak hareketlilik ya çok küçük yâda hiç olmamaktadır. Bundan dolayı rastgele erişim periyodunun ana fonksiyonu düğüm eklenmesine ve silinmesine izin vermektir. Zaman senkronizasyonu periyot süresince yapılabilmektedir.

Rastgele erişim periyodunda bütün düğümler ya iletim yâda alıcı durumda olmaktadırlar, böylece komşuluk güncellemelerini gönderebilmekte ve komşularından güncellemeleri alabilmektedirler. Rastgele erişim periyodu sırasında sinyal paketleri çarpışmadan dolayı kaybolabilmektedirler. Buda düğümler arasında tutarsız komşuluk bilgilerine yol açmaktadır. Tutarlı komşuluk bilgilerinin güvenliğini garanti etmek için rastgele erişim periyodunun uzunluğu ve sinyal paketlerinin tekrar gönderim sayısı buna göre ayarlanmaktadır [14,26]. NP komşuluk bilgilerini rastgele erişim periyodu sırasında küçük sinyal paketlerinin değiş tokuşu ile toplamaktadır. Bu protokolde, düğümler boşta iken anahtarlama işlemi gerçekleştirilerek ve paket çakışması önlenerek enerji tüketimi azaltılmaya çalışılmıştır. TRAMA protokolü diğer benzer protokollerle karşılaştırıldığında gecikme daha fazla olmaktadır.

2.3.13. Aloha ile öntakı örnekleme

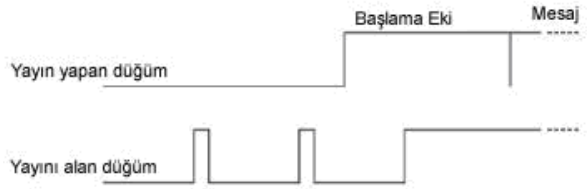
Aloha ile öntakı örnekleme protokolü, ALOHO protokolünü ve öntakı örneklemesi tekniğini birleştiren bir protokoldür. Öntakı örnekleme tekniğinin amacı, kanal boşken alıcının çoğunlukla uykuda kalmasına izin vermektir. Bu her paketin önünde belirli bir uzunlukta bir öntakı iletimi içermektedir. Bir alıcı belirli sürelerde periyodik olarak uyanır ve kanaldaki aktiviteyi kontrol eder. Eğer kanalı boş bulursa, alıcı tekrar uykuya dalar. Eğer bir öntakı tespit edilirse, alıcı uyanık kalır ve paket alınana kadar dinlemeye devam eder.

Aloha ile öntakı örneklemesinin özellikleri bir araya getirildiğinde, ortamı gereksiz dinleme için harcanan zamanı azaltılabilmektedir. Uzun iletimden dolayı karşılama uzunluğu artmaktadır. Ayrıca çarpışma ihtimalide artacaktır. Herbir mesaj bir öntakı tarafından önceliklendirilmektedir. Yani gönderen düğüm, yapılmak istenen iletişimin başarılı olup olmadığını anlamak için alıcı düğümden bir cevap bekler. RX'ten TX'e dönüştürme süresinden sonra, bir mesajın başarıyla alınmasında, hedef düğüm geriye kabul mesajı gönderecektir [27].

Gönderilen pakette bir çakışma oluşursa belirli bir süre sonra tekrar gönderim gerçekleştirilir. ALOHO protokolünün en büyük dezavantajı ortamın gereksiz yere dinlenmesidir. Burada oluşacak enerji kaybının önlenmesi için düşük güç dinleme tekniği kullanılmaktadır. Gelen mesajların örneklenebilmesi için mesajda bulunan başlık, sonraki mesajların alıcısını belirleyen bir öntakı kullanmaktadır. Gönderilen öntakı alıcı taraf tarafından duyulmazsa, sonraki örnek gelene kadar radyo kapatılmaktadır.

Öntakı örnekleme tekniği, kanal meşgul olduğunda haber veren Genie'yi uygulamanın tek yoludur. Diğer yol, sürekli dinlemede olan ikinci bir uyanık alıcıya sahip olmak olabilir. Bu ikinci alıcı çok basit olmalı ve bu nedenle sabit kullanımına izin verecek şekilde çok az güç harcamalıdır. Ortamdaki trafik tespit edildiğinde, ana alıcı uyanık olmaktadır. Dönüştürücü tarafında, mesajın başlamasında ana alıcı

uyanık olabilsin diye mesaj iletimi önceliklendirilmelidir. Bu protokolün örnekleme mimarisi Şekil 2.6'da gösterilmiştir [27]. Bu protokolün, düşük trafik koşulları altında bulunan uygulamalar için daha uygun olduğu görülmektedir.



Şekil 2.6. Aloha ile öntakı örnekleme mimarisi

2.3.14. X-MAC

X-MAC, B-MAC protokolünden esinlenmiştir. X-MAC enerji tüketimini ve gecikmeyi azaltmak için kısa öntakı uygulamakta ve düşük güç dinleme işlemine olanak sağlamaktadır. Öntakıda hedefin adres bilgilerini bulundurmakla hedef olmayan alıcı düğümlerin hızlıca uykuya geçmesi amaçlanmıştır. Ayrıca, hedef alıcı düğüme uzun öntakıyı kesme imkânı sağlamakta ve düğüme uyanır uyanmaz tetiklenmiş öntakı kullanma imkânı tanımaktadır.

Kısa tetiklenmiş öntakı yaklaşımı bütün öntakının tamamlanması için harcanan enerji ve zamanı azaltmaktadır. X-MAC kısa öntakı kullanmakta ve bu öntakıların her biri hedef adres bilgisi içermektedir. Böylece düşük güç dinleme probleminden kaçınmakta ve hedef olmayan alıcılardaki enerji korunumunu sağlamaktadır. Ayrıca kısa öntakı kullanıldığından gecikmeyi de azaltmıştır [14,28]. Tek yöne yayın esnasında B-MAC'e göre gecikme, verim ve güç tüketimi alanlarında daha iyi sonuçlar vermektedir. İstem dışı dinlemeden doğan enerji kayıpları azaltılmıştır. Her noktaya yayında B-MAC'le aynıdır.

X-MAC protokolünün KAA'lardaki görev çevrimlerindeki tasarım amaçları enerji verimliliği, basit, düşük ek yük, dağıtık uygulama, düşük gecikme, yüksek üretilen iş miktarı olarak açıklanabilmektedir [28]. X-MAC düşük güç iletişimi olarak

adlandırdığı yöntemde, kısa öntakı yaklaşımıyla, düşük güç dinleme tekniğinin avantajlarını beraber kullanarak enerji verimliliğini sağlamaya çalışmaktadır.

2.3.15. Z-MAC

Z-MAC(Zebra Medium Access Control-Zebra Ortam Erişim Protokolü) hibrid yapıda olan bir MAC protokolüdür. Z-MAC protokolü algılayıcı düğümlere zaman dilimi atamaktadır. Düşük tarfik yoğunluğunda CSMA gibi, yüksek trafik yoğunluğunda TDMA gibi davranmaktadır. TDMA'nın aksine, algılayıcı ağlarda görülen dinamik topoloji değişimlerine karşı ve zaman senkronizasyon kararsızlığına karşı dayanıklıdır. Ayrıca CSMA'nın aksine çok düşük ek yüklü gizli terminalleri idare etmektedir. DRAND eski ölçeklenebilir kanal planlama algoritması kullanılmaktadır. DRAND RAND'ın ilk dağıtık uygulamasıdır [29]. DRAND merkezi kanalın yeniden kullanımı için kullanılan bir planlama algoritmasıdır. Uygulanan planda iki hop komşuların aynı dilim numarasına atanmadığından emin olunması gerekmektedir. Dilim atamasından sonra, her düğüm kendi atanmış dilimini periyodik olarak önceden belirlenmiş periyodta yeniden kullanır. Dilim başına birden fazla dilime sahip olabilmektedirler. Çünkü DRAND iki hoptan fazla ayrılan iki düğüme aynı dilime sahip olma imkânı tanımaktadır.

CSMA'da olduğu gibi bir düğüm bir dilimde iletimden önce, her zaman hattı kontrol eder ve kanal temizse paket gönderir. Buna rağmen dilimin sahibinin kanal üzerinde ötekilere göre önceliği bulunmaktadır. Bu öncelik başlangıç geriçekilme periyodu ayarlanarak uygulanmaktadır. Daha yüksek önceliğe sahip düğüm daha kısa geriçekilme periyoduna sahip olandır. Z-MAC'de bir düğüm açık bir şekilde ağ çekişmesinin o anki durumuna bağlı olarak iki operasyon modu arasında değiştirme yapabilmektedir [14,29]. Düşük çekişme altında sahibi olmayanlar düşük öncelikli dilim için rekabet etmeye izin verilir. Bu mod düşük çekişme seviyesi olarak adlandırılır. İki hop üzerinde yüksek çekişme altında gizli terminal problemi ortaya çıkabilmektedir. Düğüm daha fazla veri çekişmesi meydana gelirse modunu yüksek çekişme seviyesi olarak ayarlamaktadır. Bu modta dilime sahip olmayanlara karşı

gizli terminal olarak hareket edilmez. Z-MAC iki hop komşuluktaki göndericiler arasında sadece yerel saat senkronizasyonuna ihtiyaç duymaktadır. Z-MAC protokolünün B-MAC protokolüne benzer avantaj ve dezavantajları bulunmaktadır. Z-MAC protokolünün dezavantajı ise performansı CSMA'nın gerisinde olmasıdır. Diğer protokollerle kıyaslandığında Z-MAC protokolünün bir kaç işlem ve bellek kaynağına gereksinimi bulunmaktadır.

2.3.16. CMAC

CMAC(Convergent Medium Access Control-Yakınsak Ortam Erişim Kontrolü) düşük görev çevrimli yakınsak MAC protokolüdür. Geleneksel uyanma planı yaklaşımı, ya periyodik senkronizasyon mesajı yada herhangi bir senkronizasyon için yüksek paket ulaştırma gecikmesine sebep olmaktadır. CMAC düşük gecikmeyi desteklerken senkronizasyon ek yükünden kaçınmaktadır. Trafik olmadığı zaman sıfır iletişimi kullanmasından dolayı, CMAC işlemlere çok düşük görev çevriminde izin vermektedir. Trafik oluşursa, CMAC düğümleri uyandırmak için birinci olarak her yöne yayını kullanır ve senkronize olmayan her yöne yetersiz yol görev çevriminden tek yöne en uygun yol görev çevrimine yani senkronize bir plana yakınsanır. İletilecek bir paket olmadığı zaman CMAC, BMAC protokolüne benzer şekilde senkronize olmayan uyku planı kullanmaktadır. Eğer gönderici kabul edilebilir yönlendirme metrikleri ile bir düğüme paket iletimi yapabilecek ise CMAC her noktaya yayının ek yükünden kaçmak için her noktaya iletimden tek yöne iletime yakınsanır [30].

CMAC protokolünde uzun öntakı yerine saldırgan RTS kullanılarak uzun öntakılar parçalanmakta ve birçok RTS paketi şekline çevrilmektedir. RTS paketleri uzun öntakı kullanmamakta ve bu uzun öntakıları sabit kısa şekilde parçalayarak alıcıların geriye CTS paketi yollamasına izin vermektedir. CMAC protokolü, kanal belirlemesi için çift kanal denetimini kullanan saldırgan RTS, ileticiyi çabuk bulabilmek için her noktaya yayın ve her noktaya yayının ek yükünü azaltmak için paket iletimini yakınsama şeklinde 3 tane bileşen içermektedir [30].

2.3.17. O-MAC

O-MAC(Organized Medium Access Control-Organize Ortam Erişim Protokolü) protokolü SMAC protokolünden esinlenmiştir. O-MAC protokolünün amacı ağın ömrünü ana enerji tüketim nedenleri olan çakışma, istem dışı alım ve gereksiz dinlemeden kaçınarak uzatmaktır. O-MAC çakışma tabanlı bir protokoldür. Bu protokolün tasarımı 2 ana fikirden oluşmaktadır. Birinci olarak çekişen komşu düğümlerin olabilecek çakışmasını önlemek için CSMA protokolü üzerinde bölgesel olarak zamanı belirli algoritma benimsenmiştir. İkincisi ise iletimin olduğu çevrede bulunan ve gönderilen veri ile ilgisi bulunmayan düğümler bu iletim esnasında uyku durumuna geçebilirler ve uyku durumuna geçmeden tüm çevredeki diğer düğümleri bilgilendirerek onların kendine atık paket yollamasını bu süre içinde önleyebilirler. O-MAC protokolünde iki tane yeni kontrol paketi sunulmaktadır ve bu düğümlerin bütün düğümler için kanal rezervasyonunu doğrulamaktadır [31]. Bu sayede RTS ve CTS paketlerinin çakışmasından dolayı uyku moduna geçen düğümlerin bu işlemini önlenmektedir.

O-MAC protokolünde bir iletimin çevresinde bu iletişime gereksinim duymayan düğümlerin uyku moduna geçmeleri ile enerji korunumunu gerçekleştirilmektedir [31]. Hiç hareketlilik olmadığı veya çok az olduğu varsayılan bir ortamda algılayıcı düğümler çevrelerindeki diğer düğümler hakkında bilgi edinebilmektedirler. Buda algılayıcı ağ kurulurken başarılabilmektedir. Bu aşamada her düğüm komşularının bir tanımlayıcısını(id) elde edebilmekte ve bunları tanımlayıcıya bakarak sıralayabilmektedir. Çevredeki herhangi bir düğüm iletim ve alım durumlarında kanala erişim için düğüm tanımlayıcı tabanlı bir plan algoritmasına uymaları gerekmektedir. Bu tanımlayıcı, algılayıcı düğümün MAC adresi için bir örnek veya herhangi başka bir tanımlayıcı olabilmektedir.

Tüm komşularının N(ni) listesine sahip olan bir ni(i=id) düğümün kanala erişimi, bütün zamanda herhangi bir veriye sahip değilse, SIFS süresince serbest olacaktır. Bu zaman geçtikten sonra, komşuluk listesindeki bir sonraki düğüm kanala erişmeye

çalışacaktır. Bu amaç için klasik RTS/CTS kontrol paketlerinin yerine, iki tane yeni kontrol paketi tanımlanmıştır. Bu kontrol paketleri plan modunu aktif hale getirmektedir. Birinci kontrol paketi OTS(Order To Sleep-Uyku Talimatı) olarak adlandırılmış ve veri paketi iletecek düğüm tarafından veya bu veri paketini alacak alıcı tarafından gönderilmektedir. Bir kanala erişmek üzere iletici tarafından bir OTS gönderildiği zaman, paket komşuların listesini ve peşi sıra gelen verilerin sürelerini içerir. Bu şekilde, gönderici komşularının sayısını ve OTS'nin alıcılarının uyku moduna geçme taleplerini duyurmasının hangi sırada olacağını belirtmiş olur. Bu yayınlama işlemi uyku modunun periyodunun süresini içeren NTS (Node To Sleep-Düğüm Uykuya) paketi gönderilerek yapılır [31].

2.3.18. FLAMA

FLAMA(Flow-Aware Medium Access Protocol-Akış Farkında Ortam Erişim Protokolü) enerji verimliliğini paket almaya hazır olmayan düğümün gereksiz dinleme, veri çakışması ve iletimi engelleyerek başarmaktadır. Bu protokol ortama erişim planlarının uygulama tarafından sergilenen trafik akışına uyarlamaktadır. FLAMA yeterince basit olduğundan sınırlı işlem, bellek, iletişim ve güç yeteneklerine sahip düğümler tarafından çalıştırılabilir. FLAMA algılayıcı ağ uygulamalarında güçlü trafik öngörüsünü sağlayabilmek için kullanılan plan tabanlı bir MAC protokolüdür. Uygulamaların trafik karakteristiklerini açıkça belirtme veya her düğüm için trafik tahmin tekniklerini kullanarak trafik bilgileri belirlenebilmektedir. Eldeki uygulamaya bağlı olarak, trafik tahmini nispeten daha kolay olmaktadır [32]. Örneğin periyodik veri toplama baz istasyonunda konumlanan ve bütün ilgili düğümleri kapsayan bir ağaç yığını üzerinde veri akışı oluşturur. Veri gönderildiği zaman her düğüm baz istasyonuna doğru sonraki hop'a gelen veriyi aktarır. Bu bilgi düğümlerin iletişimi için bir sonraki hopta yer alan düğümün belirlenmesinde kullanılabilmektedir.

FLAMA akış kavramını uygulamaların trafik modellerini karakterize etmek için kullanır. FLAMA iletim planını belirlemek için akış tabanlı trafik bilgilerini kullanır,

hem de bu bilgileri düğümlerin ne zaman alıcı modta olacağına veya düşük güç uyku durumuna geçişini belirlemek için kullanır [32]. FLAMA'nın ana özelliği, enerji verimli kanal erişimi için basit uyarlamalı trafik ve dağıtık seçim planı kullanmasıdır. Bu seçimi gerçekleştirmek için 2 hop komşuluk ve komşuluk akış bilgilerine gereksinim duymaktadır. İki hop komşuluk bilgilerini kullanmak FLAMA protokolünü ölçeklenebilir yapmaktadır. Veri ve sinyalleşme için tek kanal varsayımı yapılmaktadır. Fakat FLAMA birçok kanal kullanımına kolayca adapte olabilmektedir. Kanala erişim rastgele erişim sırasında çakışma tabanlı, planlı erişim periyodu sırasında zaman dilimli şekilde olmaktadır. Rastgele erişim sırasında komşu keşfi, zaman senkronizasyonu ve tam trafik bilgileri değiş tokuşu gerçekleştirilmektedir. Veri iletişimi planlı erişim sırasında olmaktadır. Rastgele erişim periyotlarının kullanımı FLAMA protokolüne ağdaki trafik değişikliklerine ve topolojiye uyarlama izni vermektedir. FLAMA planlı erişim periyotları sırasında tam planı duyurma gereksinimi yoktur [32].

Alternatif olarak, uygulamaya özel trafik bilgilerinin çalışan uygulamanın belirli trafik modelini ve akışlarını yansıtmak için rastgele erişim esnasında düğümler arasında değiş tokuş yapılır. Buda FLAMA'nın değişen trafik davranışlarına ve topolojiye adaptasyonuna izin verir. FLAMA akış bilgilerini her düğüm için iletim planı kurmak için kullanır. Ek olarak, FLAMA düğüm tarafından oluşturulan trafik miktarına bağlı olarak düğüme zaman dilimi vererek trafik uyarlanırlığını başarmaktadır. Daha fazla veri yönlendiren veya üreten düğümler için daha fazla zaman dilimi kullanılmaktadır [32].

2.3.19. ER-MAC

ER-MAC KAA için karma bir MAC protokolüdür. ERMAC tasarımı TDMA ve CSMA protokollerinin karması şeklindedir. Bu sayede trafik ve topoloji değişiklikleri için esnek bir uyarlama sunabilmektedir. TDMA yaklaşımında çakışma olmayan zaman dilimi planını benimsemektedir. Düğümler kendilerinin planlanan dilimleri için uyanırlar, fakat diğer durumlarda güç korunumu için uyku moduna

geçerler. Acil durum kontrolüne katılan düğümler MAC davranışlarını değiştirebilirler. Bunu da yüksek ulaştırma oranı ve düşük gecikme için TDMA dilimlerinde çakışmaya izin vererek gerçekleştirmektedirler. ER-MAC protokolü düğümlerin ağa girmesi ve çıkması için senkronize ve seyrek dilim yapısı sunmaktadır. ER-MAC protokolü şu iki faktör üzerine yoğunlaşmıştır [33]. Bunlar;

- ER-MAC büyük hacimli trafik ile başa çıkmak için TDMA dilimlerinde çakışmaya izin vermektedir. Bu plan yüksek ulaştırma oranı ve düşük gecikme için enerji verimliliğini arttırmaktadır. ER-MAC düşük öncelikli paketleri yüksek öncelikli paketlerden ayırmak için iki öncelik kuyruğu tutmaktadır.
- ER-MAC düğümlerin kendi planlarını yerel olarak değiştirebilecekleri seyrek ve senkronize dilim yapısı sunmaktadır. Bu düğümlere ağa kolayca katılma veya çıkma izni vermektedir.

ER-MAC protokolü başlangıçta iletişimi CSMA/CA ile rastgele erişim mekanizmasını kullanarak yapmaktadır. Başlangıç aşaması sırasında veri toplama ağacı ve TDMA planları oluşturulur. Topoloji keşfinde baz istasyonu basit akış mekanizmasını kullanarak ağaç yapısını başlatır. Bu protokolün topoloji keşfinin amacı yalnızca yönlendirme ağacı kurmak değil, ağaçtaki komşuları ve değişiklikleri izlemedir. Baz istasyonu sekme sayısı, yeni üst üye id'si ve eski üst üye id'si parametrelerini içeren topoloji keşif mesajını oluşturur [33]. Bu mesaj düğümün ileriki alt üyelerini bulmak, üst üyeye cevap verme ve üst üyeyi değiştirmek istediği zaman önceki üst üyesini bilgilendirme işlemlerini yapabilmeleri için her noktaya yayın şeklindedir. Bu aşamada, her düğüm baz istasyonuna olan sekme sayısı, üst üye id'sini, alt üye listesini ve bir hop komşu listesini tutmaktadır.

TDMA dilim ataması aşamasında düğümler dilim ataması ve plan değiş tokuşu yaparlar böylece 2 hoplu komşuluk içerisindeki 2 düğüm aynı dilimi kullanmamış olmaktadır. TDMA dilim ataması aşağıdan yukarıya yaprak düğümlerin(alt üyesi olmayan) dilim atamasını başlattıkları bir yaklaşım takip eder. Yaprak düğümlerden dilim ataması başlatma, mesaj akışını baz istasyonuna veren bir iletim planlamasına

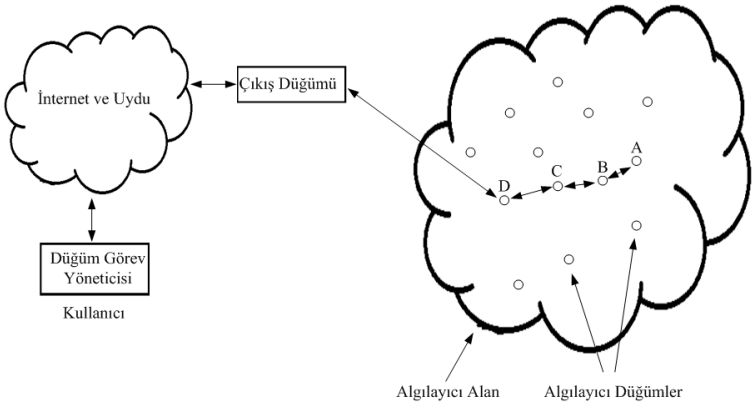
sahip olmalıdır. Baz istasyonu hariç yaprak olmayan düğümler kendi verisini göndermek için tek yönlü dilim, alt üyelerin verisini iletmek için bir çok dilim ve alt üyelerini senkron yapmak için her noktaya yayın dilimi atamadan önce bütün alt üyelerin planlarını bekler. Bu dilim atama aşaması, baz istasyonu bütün alt üyelerinden plan bildirme mesajı aldıktan sonra biter. Baz istasyonu ilk senkronizasyon mesajını göndererek TDMA'ya geçiş yapar. Alt üye mesajı aldığında TDMA'ya geçer ve kendi çocuklarını her noktaya yayın dilimi kullanarak senkronize eder [33].

2.3.20. AEEMAC

AEEMAC(Adaptive Energy Efficient Medium Access Cotrol Protocol for Wireless Sensor Networks-Kablosuz Algılayıcı Ağlar için Uyarlamalı Enerji Verimli Ortam Erişim Kontrol Protokolü) protokolü S-MAC protokolünden esinlenmiştir. Kablosuz algılayıcı ağlarda en fazla enerji tüketiminin olduğu alanlardan bir tanesi olan boşta dinleme olayı SMAC protokolü gibi AEEMAC protokolünde de görev çevrimi boşta dinlemeden kaçınılmak için kullanılmaktadır. Bunun yanında bu protokol MAC katmanında daha fazla enerji verimi elde etmek için üç ek iyileştirme kullanmaktadır. Bu iyileştirmeler uyarlamalı uyku ve kanalın tekrar kullanılması, birleştirilmiş SYNC-RTS kontrol paketinin kullanımı, iki yönlü veri ulaşımı içerisinde birleştirilmiş ACK-RTS kontrol paketini kullanmıştır [34].

3. KABLOSUZ ALGILAYICI AĞLAR

Düğümler kendi aralarında iletişimi birbirleri üzerinden sağlamakta ve gönderilen veri çıkış düğümüne (Baz) gelmektedir. Baz istasyonu direk bir cihaza bağlı olacağından buradan gelen veriler istenilmesi durumunda internet ve uydu aracılığıyla gerekli yerlere gönderilebilir. Şekil 3.1'de algılayıcı ağların genel iletişim durumu gösterilmektedir [35].



Şekil 3.1. Algılayıcı ağlardaki iletişim

Kablosuz algılayıcı ağların tasarımında dikkat edilmesi gereken bazı durumlar şu şekilde açıklanabilir;

- Algılayıcı ağlarda bulunan bazı algılayıcı düğümler enerjilerinin tükenmesi veya bazı başka nedenlerden dolayı hata verebilirler. Dolayısıyla bir veya bir kaç algılayıcı düğümün kapanması durumunda bile ağın işlevselliğini devam ettirmesi gerekmektedir [36]. Ağda oluşabilecek hatalara karşı esnek bir tasarımın bulunması gerekmektedir.
- Algılayıcı ağlar ve kablosuz ağlarda en önemli problemlerden biri olarak karşımıza çıkan ölçeklenebilirlik, ağın var olan topolojisinin değişmesi ve bu topolojiye yüzlerce yeni düğümün eklenmesi durumu karşısında çok önem kazanmaktadır. Kısaca, bir ağa ne kadar düğüm eklenirse veya çıkarılırsa her durumda ağın fonksiyonelliğinin ve veriminin devam etmesi ağın ölçeklenebilirliği ile doğru orantılı olmaktadır [35].

- Algılayıcı ağlarda kullanılan düğümlerin sayısının çok fazla olmasından dolayı maliyet bu sayı ile doğru orantılı olarak artmaktadır. Bu yüzden kullanılan düğümlerin maliyetlerinin az olması ve kullanılan düğümlerden maksimum faydayı elde etmek ortaya çıkacak maliyeti azaltmaktadır [36]. Buda hem enerji verimliliği hem iletişimin düğümler üzerinden adaletli ve devamlı şekilde yapılması ile mümkün olabilmektedir.
- Algılayıcı ağlarda topoloji oluşumu algılayıcı düğümlerin rastgele atılması veya dağıtılması yolu ile gerçeklenmektedir. Bu yüzden farklılıklar gösterse bile topolojide bulunan düğümler arasındaki uzaklıklar ortalama 2-3 metre arasında olmaktadır[36]. Topolojiyi oluşturma aşamasında meydana gelecek atılma veya dağıtıma göre ortalama metreküpe 20 düğümde düşebilmektedir. Dolayısıyla topolojiler farklı olduğundan dolayı kablosuz algılayıcı ağların oluşturulacağı alan ve metrekareye düşecek algılayıcı düğüm olasılıkları iyi değerlendirilmelidir.
- Algılayıcı ağlardaki düğümlerin boyutlarının küçük olmasından ve belirli sınırlarda veri iletişimi yapabileceğinden güç kaynakları sınırlı olmaktadır. Algılayıcı ağların oluşturulduğu yerlere göre bazen bu düğümlere ulaşmak ve enerji kaynaklarını değiştirmek veya şarj etmek mümkün olmamaktadır. Bu gibi olaylardan dolayı algılayıcı düğümün enerjisinin azalması ağın ömrünü etkilemektedir ve tasarımlar bu olaylar dikkate alınarak yapılmalıdır[37].

3.1. Kablosuz Algılayıcı Ağlarda Ortaya Çıkan Enerji Tüketimi Problemi

Enerji tüketimi ile ilgili yapılan çalışmalar genelde ağ katmanı ve veri bağı katmanı üzerinde gerçekleştirilmektedir. Ağ katmanında enerji tüketimini azaltmak için yapılan çalışmalar genel olarak yönlendirme konusu üzerinedir.

Veri bağı katmanının sorumlu olduğu iki önemli olay ortam erişimi ve hata kontrolüdür. Kablosuz algılayıcı ağlarda MAC protokolünün en önemli görevleri ise birinci olarak algılayıcı ağ alt yapısının oluşturulması, ikinci olarak kaynakların düğümlere eşit olarak paylaştırılmasıdır.

Günümüzde kablosuz ağlarda kullanılan MAC protokolleri algılayıcı ağlarda direk olarak uygulanamamaktadır. Dolayısıyla algılayıcı ağların güç kaynaklarının az olması ve algılayıcı ağlarda bulunan düğüm sayılarının diğer kablosuz ağlardakinden daha fazla olması göz önünü alındığında algılayıcı ağlardaki en büyük problemin enerji tüketimi olduğu ortaya çıkmaktadır [38,39].

Algılayıcı ağlarda düğümlerin enerji kaynaklarının değiştirilemeyeceği veya şarj edilemeyeceği konumlarda bulunması durumunda, olmazsa olmaz koşullardan biri, düğümlerin en az enerji tüketimini gerçekleştirmesidir. Çünkü konumuna göre bir veya daha fazla düğümün enerjisinin tükenmesi demek o ağın ömrünün tükenmesi anlamına gelmekte ve hem harcanan zaman hem maliyet açısından sorunlar çıkarmaktadır. Düğümlerin sahip oldukları güç kaynaklarını en verimli şekilde kullanmaları için alınabilecek bazı önlemler şu şekilde sıralanabilir [38];

- Her düğümün kapsama alanının sınırlanması. Çünkü bir düğümün uzak mesafelere veri gönderimi için harcayacağı enerji yakın mesafeye harcayacağı enerjiden daha fazladır.
- Algılayıcı düğümlerin komşu düğümler ile iletişime geçmesi ve bu iletişim esnasında herhangi bir iletişim olmaması durumuna göre düğümlerin kendilerini kapatmaları veya düşük güç harcayacakları bir konuma geçmeleri gerekmektedir.

Dolayısıyla enerji verimliliği veri bağı katmanında önemli bir sorun olarak karşımıza çıkmaktadır [38-39].

3.2. TinyOS

TinyOS işletim sistemi algılayıcı düğümler için geliştirilmiş bir işletim sistemidir ve sınırlı kaynaklara sahip algılayıcı düğümler için uygun ve verimli bir programlama amaçlamaktadır. TinyOS nesC dili ile yazılmıştır ve gerçekleştirilmek istenen uygulamaların bu dille yazılması gerekmektedir [40]. Burada düğümler ve baz

istasyonları arasındaki uygulamalar nesC dili ile yazılırken, algılayıcı ağ ile bilgisayarlar arasındaki iletişim için gerekli uygulamalar genellikle Java ile yapılmaktadır.

TinyOS açık-kaynak kodlu bir işletim sistemidir. Bu sayede uygulama için gerekli kod sayısını azaltarak algılayıcı ağlar için hızlı ve uygun bir sistem olmuştur. TinyOS bileşen kütüphanesi ağ protokollerini, dağıtık hizmetleri, algılayıcı sürücülerini ve veri elde etme araçlarını içermektedir. TinyOS'un olay tabanlı çalışması sayesinde iyi bir güç yönetimi sunmakta ayrıca ölçeklenebilirliğe izin vermektedir.

TinyOS geleneksel anlamdaki gibi bir işletim sistemi değildir, gömülü sistemler için bir programlama çatısı ve bileşenlerden oluşan bir sistemdir. Uygulama kodunun çok düşük belleğe ihtiyaç duyması önemli bir noktadır. Buna ek olarak TinyOS tasarımında bir dosya sistemi yoktur [41].

TinyOS işletim sisteminde birçok bileşen bulunmaktadır ve en güzel tarafı ise bu bileşenlerin gerçekleştirilmek istenen uygulamalara göre eklenip çıkarılabilmesidir. TinyOS sağladığı bu esneklik sayesinde bileşenler isteğe göre tanımlanabilmekte ve kullanılabilmektedir. Ayrıca gerçekleştirilmek istenen uygulamalar için geliştiriciye kolaylık sağlayan birçok programlama ara yüzü sağlamaktadır [41,42].

TinyOS işletim sisteminin geliştirildiği dil olan nesC dili aynı zamanda geliştirilen uygulamalar içinde kullanılmaktadır. nesC dili bileşen tabanlı olmasından dolayı olay tabanlı mekanizmaya sahip sistemlerin geliştirilmesi için çok uygundur. NesC dilinde kullanılan bileşenlerin her biri kendine özgü durum bilgilerini tutmakta ve o bilginin kullanılması işlemini gerçekleştirmektedirler. Daha önceden bahsedildiği gibi sunulan ara yüzler vasıtasıyla bileşenler birbirleriyle ilişki kurmaktadırlar. Bu tip ağlardaki sınırlamalardan dolayı kullanılacak bileşenler ve aralarındaki ilişkiler başta belirlenmektedir[40]. Yanı dinamik bir yapı sunamamaktadır. TinyOS işletim sisteminde tek yığın ortamı oluşturulmakta ve buradaki tüm işletim bu ortam üzerinden halledilmektedir. Bu ortam sayesinde kullanılan fonksiyonların işlevi

bitinceye kadar çalışmaktadır. Bu sınırlı bir yapıya sahip olan algılayıcı ağlarda kullanım kolaylılığı ve verimlilik sağlamaktadır. TinyOS işletim sisteminde modül olarak isimlendirilen bileşenler sağladıkları ara yüzlerin C dili ile uygulanması işlemini kendilerinin kullandıkları ara yüzler ile başarmaktadırlar. Burada diğer bileşenleri bağlayan bileşenlere yapılandırıcı ismi verilmektedir. TinyOS işletim sisteminde bulunan bileşenler ve ara yüzlerin isimleri, gerçekleştirim dosyalarının isimleriyle aynı olmalıdır. TinyOS'ta bir bileşen başka bileşenin komutlarını kullanabilmekte, komutları kullanılan bileşen de uygulama anında ortaya çıkan olayları sinyaller ve komutları kullanan bileşeni meydana gelen olaylardan haberdar edebilmektedir [43].

TinyOS işletim sistemi kullandığı nesC dili sayesinde uygulama geliştirmeyi nesneye yönelik programlama da olduğu gibi kolaylıkla yapılmasına olanak sağlamaktadır. Diğer taraftan sağladığı diğer bir özellik ise gereksiz bellek kullanımını ve ek katmanların yarattığı verimsizlikten gerçekleştirilen uygulamaları kurtarmayı amaçlamaktadır. Dolayısıyla TinyOS işletim sistemi durağan bir bellek tahsisi mekanizmasını kullanmaktadır. Kısaca daha derleme anında gerçekleştirilecek uygulamanın bellek gereksinimi belirlidir uygulama bitinceye kadar bu değişmez. Kısaca TinyOS işletimin sisteminde bir uygulamanın her şeyi derleme anında belirlenir ve uygulama sonlanıncaya kadar değişmez [44].

Bileşenler 3 tane hesaplama kavramı içerirler. Bunlar; komutlar, olaylar ve task'lardır. Komutlar ve olaylar bileşenler arası iletişim için birer mekanizmadır. Task'lar ise bileşen içi işlerde kullanılmaktadır. Komut geleneksel olarak bileşenden bir servisi çalıştırmasını ister, örneğin algılayıcı okumanın başlaması gibi. Bir olay ise hizmetin tamamlandığı sinyalidir [45,46]. Olaylar asenkron olarak sinyalleşebilir, örneğin donanım kesmelerinden dolayı veya mesaj varışlarından dolayı. Bir hesaplamanın acilen çalıştırılması yerine, komutlar ve olay yakalayıcılar bir task gönderebilirler, bu fonksiyon TinyOS planlayıcısı tarafından sonra çalıştırılabilir. Bu komutların ve olayların duyarlı olmasına izin verir, uzun hesaplama task'larını erteleyerek acil dönebilirler.

Tasklar önemli işlemleri çalıştırırken, onların temel çalışma modeli çalışmayı tamamlar. Bunu sonsuz çalışma yerine yapmaktadırlar. Bu da tasklara daha az yük imkânı sunar. Task'lar bileşen içinde iç eşzamanlılığı sunar ve sadece o bileşenin durum bilgisine erişirler [42,47]. TinyOS planlayıcısı FIFO(First In First Out-İlk Giren İlk Çıkar) mantığını kullanır. Geliştiricinin bir uygulamayı bileşenleri yazarak ve bunları diğer TinyOS bileşenlerle bağlayarak gereken servislerin uygulamasının sağlanmasını gerçekleştirirler.

3.2.1. TinyOS programlama

TinyOS işletim sistemi, kütüphaneler ve uygulamaların hepsi nesC ile yazılmıştır. nesC yeni bileşen tabanlı bir dildir. nesC dili birincil olarak gömülü sistemlere yöneliktir, örneğin algılayıcı ağlar gibi. nesC C diline benzerdir, fakat TinyOS'in eşzamanlı modelini desteklemekte, yanı sıra sağlam gömülü sistem ağları içinde yazılım bileşenlerinin yapılandırılması, isimlendirilmesi ve bağlanmasını desteklemektedirler. Buradaki birincil amaç uygulama tasarımcılarına bileşenlerin derlenmesi için kolaylık sunulmasıdır, ayrıca eş zamanlı sistemler ve derleme zamanında geniş bir kontrol imkânı sunmaktadırlar [43,48]. TinyOS'un nesC'de tanımlanmış belirli sayıda önemli kavramı bulunmaktadır. Bunlar Çizelge 3.1'de gösterilmektedir.

Çizelge 3. 1. TinyOS/nesC kavramı

Terimler	**Açıklama**
Application	Bir TinyOS/nesC uygulaması bir veya birden fazla bileşen içerebilirler. Bu bileşenler birbirlerine çalışma zamanında birbirleriyle çalışması için bağlanırlar.
Component	Bileşenler nesC uygulamaları için temel bloklardır. 2 tip modül vardır. Bunlar; modules ve configurations'dır. Bir TinyOS bileşeni ara yüz desteği verebilir veya kullanabilir.
Module	Bir bileşen bir veya daha fazla ara yüzü kullanabilir.

Configuration	Bir bileşen diğer bileşenler ile bağlanır, bağlanılan ara yüzler bileşenler tarafından kullanılırlar (diğerleri tarafından sağlanan ara yüzler). Buna bağlama denir. Geliştiriciler bir uygulamayı modüller kümesi ile ve bu modüllerin configürasyon dosyasında bağlamasıyla oluşturabilirler. Dolayısıyla tüm nesC uygulamaları konfigurasyonun en başında uygulamada kullanılacak bilşenleri ve ilişkileri belirlerler.
Interface	Bir ara yüz 2 bileşenin etkileşiminin tam tanımlanmasını sağlamak için kullanılırlar. Bu kavram Java'dakine benzer ve bir ara yüz kod veya bağlama içermez. Basitçe ara yüz sağlayıcısının uygulaması gereken fonksiyonlar, komutlar ve talep edenlerin kullanması gereken fonksiyonlar ve olaylar içerir. Bu yönüyle javadan farklıdır çünkü Java ara yüzlerinde çağırma tek yönlüdür. NesC ara yüzlerinde iki yönlüdür. Bir bileşen için ara yüzdeki bir komutu çağırmak o ara yüzün olayını uygulamasıyla mümkün olmaktadır. Tek bir bileşen birçok ara yüze destek verebilir veya ihtiyaç duyabilir ve aynı ara yüzün birden fazla örneğini kullanabilir. Bu ara yüzler sadece bileşenlere erişim içindir.

3.2.2. NesC programlama dili

Bu bölümde nesC dilinin temel özellikleri ve kullanılan terimler detaylı bir biçimde açıklanacaktır.

Makefile: Bir uygulamayı oluşturmanın ilk basamağı makefile'ı yazmaktır. Alternatif olarak bu uygun başka bir makefile dosyasından kopyalanabilmektedir. Makefile kodu aşağıdaki gibi yazılır [48].

```
include Makefile.component
include $(TOSROOT)/apps/MakeXbowlocal
include $(MAKERULES)
```

Makefile.component: Bir sonraki aşama ise Makefile.component dosyasını oluşturmaktır. Bu dosya uygulama bileşeninin en başında tanımlanır. Kullanacağımız uygulamanın adı ve algılama tahtasının adı burada belirlenir. Algılama tahtasının referansı derleyiciye bu tahtada algılayıcı cihazlara erişmek için nesC bileşenlerini kullanacağımızı anlatır. Her algılama tahtasının kendine ait nesC algılayıcı bileşenleri vardır [48,49]. Örnek kod aşağıdaki gibidir.

```
COMPONENT=Tez
SENSORBOARD=mts310
```

Konfigürasyon dosyasının oluşturulması: Tez.nc dosyasında konfigürasyon dosyası bulunmaktadır. StdControl ara yüzü daima bir uygulama da kullanılmalıdır. StdControl arayüzü TinyOS'un temel fonksiyonelliğini sağlamaktadır. Örneğin ilk değer verme, başlama ve bitirme gibi. Aşağıda örnek bir yapılandırma kodu gösterilmektedir.

```
configuration Tez
                {
                }
implementation  {
                  components Main, TezM, TimerC, LedsC;
                  Main.StdControl -> TimerC.StdControl;
                  Main.StdControl -> TezM.StdControl;
                  TezM.Timer -> TimerC.Timer[unique("Timer")];
                  TezM.Leds -> LedsC.Leds;
              }
```

Konfigürasyondaki en son iki satır TimerC ve LedsC bileşenlerini uygulama modülüne bağlamaktadır. Bu sayede modül Timer ve LED cihazlarını TimerC ve LedsC bileşenlerinin fonksiyonlarını çağırarak kontrol edebilmektedir [43,48].

Modül oluşturma: Uygulama modülü TezM.nc dosyasında bulunmaktadır. Modül dosyası kısaca uygulamanın programlama kodunun girildiği yerdir. Yani düğüm

üzerindeki lambanın yanmasının/sönmesinin ve zamanlayıcıyı başlatmak için kodun yazıldığı kısımdır.

```
module TezM {
provides {
            interface StdControl;
         }
uses     {
            interface Timer;
            interface Leds;
         }
            }
implementation {
                   command result_t StdControl.init()
                   {
                         call Leds.init();
                         return SUCCESS;
                   }
                  command result_t StdControl.start()
                  {
                         return call Timer.start(TIMER_REPEAT, 1000);
                  }
                  command result_t StdControl.stop()
                  {
                         return call Timer.stop();
                  }

                  event result_t Timer.fired()
                  {
                         call Leds.redToggle();
                         return SUCCESS;
                  }
}
```

Modüller ve yapılandırma dosyaları arasındaki farkın sebebi geliştiriciye uygulama ile daha önceden var olan bileşenlerle daha hızlı şekilde ek bir program yazmadan

kullanılma imkanı vermesidir. Örneğin geliştirici yapılandırma dosyası ile bir veya daha fazla modülü basitçe bağlayabilmektedir.

Kullanılan ara yüzlerle ilgili önemli bir not ise alt bileşenin ilk değer verme fonksiyonu kesinlikle kullanan bileşen tarafından çağırılmalıdır. Örneğin TezM modülü Leds arayüzlerini kullanmaktadır [48]. Bundan dolayı Leds.init() kesinlikle TezM.init() tarafından çağrılmalıdır.

nesC ara yüzler arasındaki ilişkiyi belirlemek için oklar kullanmaktadır. Sağ yön okunun "->" anlamı "binds to" demektir. Sol yöne olan ok ise bir ara yüzü sağ taraftaki uygulamaya bağlar. Diğer bir deyişle bileşen soldaki bir ara yüzü kullanır ve bileşen sağ taraftaki ara yüzü destekler [48,51]. Alttaki satır bağlamaya bir örnek olarak gösterilmektedir;

```
TezM.Timer -> TimerC.Timer[unique("Timer")];
```

TezM tarafından kullanılan Timer ara yüzünü TimerC tarafından sağlanan Timer ara yüzüne bağlamak için kullanılır. Okun sol tarafındaki TezM.Timer Timer (/tos/interfaces/Timer.nc) olarak adlandırılan ara yüze referans etmektedir. Okun sağ tarafındaki TimerC.Timer ise Timer'ın uygulamasına referans eder. Unutulmaması gereken oklar daima soldaki ara yüzü sağdaki uygulamaya bağlar. unique("Timer") kodu ise kullanılan Timer örneğinin uygulamanın başka bir yerinde kullanılmasını önlemektedir.

nesC aynı ara yüzün birçok uygulamasını desteklemektedir. Timer ara yüzü bir örnektir. TimerC ise birçok zamanlayıcıyı Timer Id'yi parametre şeklinde kullanarak sunar. Bağlantıların yazılış biçimi kısaltılabilir.

```
TezM.Leds -> LedsC; kodu
TezM.Leds -> LedsC.Leds; kodunun kısa yazılmış halidir.
```

Bu örnekte leds olarak adlandırılan ara yüz tam listelenmemiştir. Hiçbir ara yüzün adı okun sağ tarafında verilmemiştir. nesC derleyicisi default olarak aynı ara yüzü okun sol tarafındakiyle bağlamaya çalışır [48].

Main bileşeni: Kavramsal olarak TinyOS uygulaması bileşenlerin toplanması ve Main olarak adlandırılan bir planlayıcıdan oluşmaktadır. Bileşenler tipiksel olarak bazı hesaplamaları veya fonksiyonları sağlar ve planlayıcı ise bu bileşenler tarafından oluşturulan task'ları çalıştırırlar. Main'in kodu aşağıdaki gösterilmektedir.

```
configuration Main {
                        uses interface StdControl;
                    }
                   implementation
                    {
                        components RealMain, PotC, HPLInit;
                        StdControl = RealMain.StdControl;
                        RealMain.hardwareInit -> HPLInit;
                        RealMain.Pot -> PotC;
                     }
```

Main bileşeni TinyOS uygulamalarında ilk çalıştırılan bileşendir. Kesin olarak, Main.StdControl.init() komutu TinyOS'da çalıştırılan ilk komuttur ve Main.StdControl.start() komutu önceki kodtan sonra çalıştırılır. Bundan dolayı bir TinyOS uygulaması Main bileşenine ve onun yapılandırmasına sahip olmalıdır. StdControl ise TinyOS bileşenlerine ilk değer verme ve başlatmak için kullanılan bir ara yüzdür [48,52].

```
interface StdControl { command result_t init(); command result_t
start(); command result_t stop(); }
```

Görüldüğü gibi StdControl 3 tane komut içermektedir; init(), start(), ve stop(). init() komutu bileşenin ilk değer verme işlemi olduğu zaman çağrılır. start() komutu bileşen başladığı zaman yani ilk defa çalıştırıldığı zaman çağırılır. stop() komutu ise

bileşen durdurulduğunda çağırılır. Örneğin cihazın gücünün durdurulmasının kontrolü için kullanılır. init() komutu birçok kez çağırılabilir ancak kesinlikle start() veya stop() komutları çağırıldıktan sonra çağırılmaz. Bir bileşen için init() çağırılırsa bütün alt bileşenlerinde init() komutu çağırılmalıdır. Timer.nc StdControl biraz daha farklıdır.

```
interface Timer {
                  command result_t start(char type, uint32_t
                  interval);
                  command result_t stop();
                  event result_t fired();
                }
```

Yukarıdaki kod parçacığında görüldüğü gibi Timer arayüzü start() ve stop() komutlarını ve fired() olayını tanımlamıştır. nesC kelimesi result_t komut veya olay tarafından geri döndürülen durum değerinin veri tipidir. Bu geri dönüş değeri 2 değerde olabilir. Bunlar SUCCESS veya FAIL şeklinde olabilmektedir. start() komutu Timer'ın tipini belirlemek için ve Timer'ın biteceği zaman aralığını belirlemek için kullanılır. Zaman aralığı birimi milisaniyedir. Geçerli tipleri TIMER_REPEAT ve TIMER_ONE_SHOT olarak adlandırılır. One-shot timer belirlenen zaman aralığından sonra sona erer. Repeat timer ise stop() komutu ile durdurulmadığı müddetçe devam eder [43,48]. Peki, bir uygulama zamanın dolduğunu nasıl bilmektedir. Zamanın dolduğu bir olay çalıştırıldığında bilinmektedir. Timer ara yüzü aşağıdaki olayı sağlamaktadır.

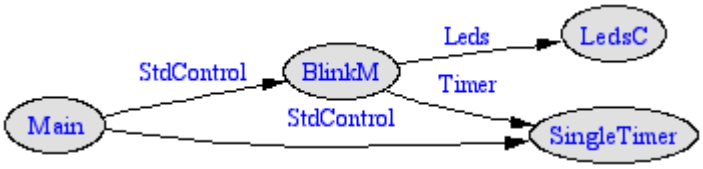
```
event result_t fired();
```

event bir fonksiyondur ve bir olay meydana geldiğinde ara yüzün sinyal üretme durumudur. Bu durumda kısaca fired() olayı belirlenen zaman aralığı geçtiği zaman sinyal üretmektedir. Bu çift yönlü ara yüze bir örnektir. Yani ara yüz sadece ara yüzün kullanıcıları tarafından komutları sağlamakla kalmamakta aynı zamanda olayların sinyal üretmesini sağlamaktadır.

Uygulamanın Bileşen Dokümanının Oluşturulması: Uygulamada kullanılan bileşenlerin grafiksel sunumu basit bir kodla elde edilebilmektedir. TinyOS kaynak kodlar metadata ve comment blokları içermektedir. nesC derleyicisi bunlar için otomatik html dokümanı oluşturmaktadır. Dokümantasyonu oluşturmak için aşağıdaki kodu kullanmak gerekmektedir [48,50].

```
make <platform> docs
```

Kod çalıştırıldıktan sonra dokümantasyon sonucu dosyada dosya adı ile oluşturulmakta ve /doc/nesdo/<platform>.docs/nesdoc/<platform>/index.html dizininde gösterilmektedir. Örnek bir Blink uygulamasının bu kod çalıştırıldığında elde edilen görüntüsü Şekil 3.2'de verilmiştir.



Şekil 3.2. Blink uygulaması dokümantasyonu

3.2.3. PowerTOSSIM

PowerTOSSIM KAAiçin ölçeklenebilir bir ortam ve her düğüm için kesine yakın enerji tüketimi tahmini sunmaktadır. PowerTOSSIM, TOSSIM'in genişletilmiş halidir ve TinyOS uygulamaları için olaya dayalı benzetim ortamı sunmaktadır. PowerTOSSIM'de TinyOS bileşenleri ile ilgili olarak belirli donanım durumları için (örneğin radyo, EEPROM, LEDs, v.s.) benzetimin çalışması sırasında her cihazın aktivitesine göre bir çıktı elde edilmektedir. PowerTOSSIM her cihaz tarafından çalıştırılan CPU çevriminin tahmini için yeni bir kod dönüşüm tekniği kullanmaktadır ve algılayıcı düğümlere benzetim imkânı sunarak bu pahalı komut-seviye işlemini elimine etmektedir. PowerTOSSIM MICA2 algılayıcı düğüm platformu üzerinde donanım enerji tüketiminin detaylı bir modelini içermektedir [45].

PowerTOSSIM yeni bir kod dönüşüm tekniği kullanarak her düğüm için CPU çevrim çalışmasını tahmin etmektedir. Son olarak, her düğümün aktivitesini detaylı olarak incelemek için donanım enerji tüketimine göre detaylı bir model sunmakta ve her düğümün enerji tüketimi konusunda verimli sonuç alınması mümkün olmaktadır [45,51]. Bu enerji modeli diğer donanım platformları içinde kolayca değiştirilebilir olma özelliği sunmaktadır.

PowerTOSSIM çok geniş ağları ölçeklemek için tasarlanmış ve Atemu benzetiminden 20 kat daha hızlı çalışmaktadır. PowerTOSSIM algılayıcı ağlar için geliştirilen ilk ölçeklenebilir benzetim programıdır ve doğru enerji tüketim verileri elde edilmesini sağlamaktadır.

PowerTOSSIM'de TinyOS işletim sistemi ve TOSSIM benzetim ortamı ele alınmıştır. TinyOS bir benzetim ortamı olan TOSSIM'i desteklemektedir. TOSSIM'de TinyOS uygulamaları bir benzetim cihazı üzerinde olaya dayalı benzetim ile doğrudan derlenmektedir. Bu tasarım ile TinyOS'un bileşen odaklı özelliğinden faydalanılmaktadır. Yani TinyOS bileşenlerinin donanıma kolay erişiminden faydalanılmaktadır. TOSSIM donanım bileşenlerini örneğin basit radyo yığını, algılayıcılar ve diğer çevre birimlerinin benzetimini desteklemektedir. PowerTOSSIM ise TOSSIM ve TinyOS bileşen modellerini enerji tüketimini takip etmek için kullanmaktadır [45,52].

Var olan benzetim ortamlarına örnek olarak ns2, TOSSIM ve Atemu verilebilir. Bu ortamlar sayesinde algılayıcı ağların davranışları detaylı olarak anlaşılabilir. Çünkü bu ortamlar çeşitli derecelerde ölçeklenebilirliği ve gerçekçiliği sağlamaktadırlar [52]. PowerTOSSIM, benzetimi yapılan düğümlerin donanım bileşenlerinin güç durumunu takip etmekte ve bu işlemi de belirli güç durum dönüşümü mesajı oluşturarak yapmakta ve bu mesajlar benzetim sırasında kayıt altına alınmaktadır. Bunda başarı ise TOSSIM'in donanım bileşenlerini yeni bir bileşen çağırarak benzetim yapmasıyla ortaya çıkmaktadır. Bu yeni bileşen Güç-Durumudur. Bu bileşen her düğüm için donanım güç durumunu izlemekte ve çalışma sırasında

bunları bir dosyaya kaydetmektedir. CPU kullanımının tahmini bununla daha çok ilişkilidir. CPU profili benzetim tarafından çalıştırılan temel bloklar sayesinde ilgili düğümün çevrim sayısının bulunması ile başarılmaktadır [45,50].

Güç-Durum bileşeninin verileri PowerTOSSIM tarafından oluşturulmakta ve bir güç modeli ile birleştirilebilmektedir. Örneğin her düğümün ve her bileşenin enerji kullanımlarına karar verebilmek için bu işlem yapılmaktadır. Bu işlem çevrimdışı araçlar kullanılarak yapılabilir. Bu sayede her düğümün her donanım bileşeni için enerji tüketimi daha detaylı elde edilebilmekte veya TinyViz ekranı kullanılarak gerçek zamanlı enerji tüketim verileri gösterilebilmektedir.

Radyo, düğümlerdeki önemli güç tüketicisidir ve radyonun durumunu doğru şekilde takip etmek algılayıcı ağlarda herhangi bir güç görüntü aracının başarılı olması için gereklidir. MICA2 platformu farklı bir radyo cipi kullanmaktadır. CC1000 radyo 3 temel durum içermektedir. Bunlar iletim, dinleme ve uyumadır. Uygulamaya göre değişmekle birlikte uyuma durumunda sabit enerji tüketimi olmakta iken dinleme ve iletim durumlarında enerji tüketimi artmaktadır [51].

CPU güç durumu için MICA2 tarafından kullanılan Atmega 128LCPU'da 7 tane farklı güç durumu bulunmaktadır. Bunlar; aktif, IDLE, ADC NOISE REDUCTION, POWER_DOWN, POWER_SAVE, STANDBY ve EXTENDED STANDBY'dır. Bir kaç TinyOS uygulaması bu durumlardan faydalanırken PowerTOSSIM'de bunların yakalanması daha anlaşılabilir olmaktadır. Varsayılan CPU aktif durumdayken yani komutları çalıştırırken ve IDLE durumdayken bir kesme gelmesini bekleme durumunda olabilmektedir. Genelde CPU çevrim tahminine bakılarak aktif durumda ne kadar zaman kaldığı belirlenebilmektedir [42]. Diğer durumda ise CPU IDLE durumunda olmaktadır.

Diğer güç durumları belirli CPU kayıt ayarları tarafından kayıt altına alınmış olur. Burada iki standart vardır ve TinyOS bileşenleri bunlardan yani HPLPowerManagament ve Snooze bileşenlerinden yararlanmaktadır. Güç durumu

için bu iki bileşenden birini çağırmak gerekmektedir. Sebebi ise CPU'nun düşük güç durumuna giriş ve çıkışını yakalamak hesaplama açısından önemlidir [42,52].

LED'lerin güç durumu için MICA2 üzerinde bulunan 3 led önemli bir güç tüketmektedir (her biri yanarken 2,2 mA). Dolayısıyla bu Led'lerin açık/kapalı durumlarının takip edilmesi önemlidir. TinyOS ledsC bileşeni bu amaç için güç durumunda çağrılmak için düzenlenmelidir.

EEPROM güç durumu için MICA2 ek olarak bir EEPROM içermekte bunu da veri kaydetme için yapmaktadır. TinyOS EEPROM modülü sayesinde düğümün kendi EEPROM'una yazma ve okumada ne kadar süre harcadığını takip edebilir. EEPROM'a yazma ve okuma farklı miktarda güç tüketmekte ve farklı zamanlarda bitmektedir. Bundan dolayı bu durumların her dönüşümü güç durumu tarafından kaydedilmektedir [42,45].

ADC güç durumu için anolog-dijital çevrimi fiziksel çevreden örnek anolog algılayıcı veri almak için kullanılmaktadır. MICA2'de ADC CPU'ya yerleşiktir ve aktifken de ayırt edilebilir miktarda güç tüketmektedir. Alternatif algılayıcı düğüm platformları harici bir ADC'yi kapsayabilmekte ve bu ADC'ler bağımsız olarak aktif veya kapalı olabilmektedirler. Bundan dolayı TinyOS ADCC bileşenini ADC aktifken ki zamanı takip etmek için düzenlemiştir[42,47].

Algılayıcılarda güç durumu ise, her düğüme bağlı olan algılayıcıların aktivasyon durumu, TinyOS'taki uygun donanım sürücü modülü düzenlenerek takip edilebilmektedir. Örneğin accelerometer algılayıcısının aktivasyon durumu güç durum modülünün TinyOS AccelM bileşenini çağırması ile yakalanması mümkündür. Basit MICA2 algılama tahtaları bir düğüme bağlandıklarında sabit miktarda enerji tüketmeleri için kullanılmalıdır. Bu yüzden photoresistor ve thermistor algılayıcılarının güç durumları güç durum modülü tarafından takip edilmelidir. MICA2 güç modeli bu algılama tahtası için sabit enerjiyi hedeflemektedir [49].

MCU (Micro Controller Unit-Mikro Denetleyici Birimi) içerisinde 6 tane güç kipi durumu bulunmaktadır. Bu uyku kipleri, MCU kontrol yazmaçlarındaki (SM2,SM1,SM0) 3 bitin değerine bağlı olarak değişmektedir. Bu üç bit uyku yazmacı olarak çalışmaktadır. SE bitinin 1 olması CPU'nun uyku kipine girmesi demektir (Bu da uyku komutlarının çalıştırılması ile mümkün olmaktadır). Tavsiye edilen ise SE bitinin uyku komutları çalıştırılmadan ayarlanması yani 1 yapılması ve uyanma işleminden sonra temizlenmesidir [42,51].

IDLE Kipi: Ne zamanki uyku yazmacının değeri 000 ise ve uyku komutları çalıştırılırsa işlemci IDLE durumuna geçer. CPU durur ancak SPI, USART, Analog karşılaştırıcı, İki telli kanal seri ara yüzü, Zamanlayıcı/Sayaçlar, Gözetleme ve kesme sistemleri işlemlerine devam ederler. Bu kip clkCPU ve clkflash'ı durdurur. Eğer ki analog karşılaştırıcı kesmesi uyanma için gerekli değilse, analog karşılaştırıcılarda bulunan ACD bitiyle, durum kontrol yazmaçları(ACSR) ayarlanarak kapatılabilir. Uyanma işlemi ise harici kesmelerle, yani örneğin zaman aşımı ve USART iletişim tamamlama ile gerçekleşir [42].

ADC NOISE REDUCTION Kipi: Ne zamanki uyku yazmaçlarının değeri 001 ise ve uyku komutları çalıştırılırsa CPU bu kipe girer. IDLE kipi ile kıyaslanırsa CPU'nun yanında SPI, USART ve Analog karşılaştırıcı ayrıca durdurulur. Ayrıca buna ek olarak clkI/O da durdurulur. Bu kip ADC için gürültü çevresini geliştirir. Uyanma işlemi ise ADC iletişiminin bitmesi dışında sadece harici bir reset, Gözcü sıfırlama, Voltaj-Düşüklüğü sıfırlama, İki telli kanal seri ara yüz adresleme kesmesi, Zamanlayıcı/Sayac0 kesmesi, SPM/EEPROM hazır kesmesi, INT7:4 üzerinde harici seviye kesmesi veya INT3:0 harici kesmesi MCU'yu bu uyku kipinden uyandırır [42,48].

POWER_DOWN Kipi: Ne zamanki uyku yazmaçlarının değeri 010 ise ve uyku komutları çalıştırılırsa CPU bu kipe geçer. Bu kipte, harici Oscillator durdurulur ancak harici kesmeler, İki telli kanal seri ara yüz adresleme ve gözcü sıfırlama aktif ise işlemine devam eder. Bu uyku kipi bütün oluşturulan saatleri durdurur. Sadece

asenkron modüllerin işlemine izin verir. CPU'nun uyanması için bir kesme geldiği zaman uyanma işlemi için değişen seviyenin düzenlenmesi gerekmektedir. Bu da belirli bir zaman aralığında olmaktadır. Uyanma işlemi ise harici bir reset, Gözcü sıfırlama, Voltaj-Düşüklüğü sıfırlama, İki telli kanal seri ara yüz adresleme kesmesi, INT7:4 üzerinde harici seviye kesmesi veya 3:0 da harici kesme durumunda gerçekleşir[42,52].

POWER_SAVE Kipi: Ne zamanki uyku yazmaçlarının değeri 011 ise ve uyku komutları çalıştırılırsa, CPU bu kipe girer. Bu kip POWER_DOWN kipiyle bir şey hariç aynıdır. Eğer Zamanlayıcı/Sayaç0 saati asenkron ise ASSR'deki AS0 biti 1 değeri şeklinde ayarlanır ve Zamanlayıcı/Sayaç0 uyku durumunda da çalışır. Uyanma işlemi ise harici bir sıfırlama yani Zamanlayıcı/Sayac0 kesmesi ile olmaktadır [42,49].

STANDBY Kipi: Ne zamanki uyku yazmaçlarının değeri 110 ise ve uyku kipleri çalıştırılırsa CPU bu kipe girer. Bu kip POWER_DOWN kipi ile aynıdır bir şey hariç, oda oscillator çalışmaya devam eder. İşlemcinin uyanması bu kipte 6 saat çevriminden sonra olmaktadır [42].

EXTENDED STANDBY Kipi: Ne zamanki uyku yazmaçlarının değeri 111 ise ve uyku komutları çalıştırılırsa CPU bu kipe geçer. Bu kip POWER_SAVE kipi ile aynıdır bir şey hariç o da oscillator burada çalışmaya devam eder. İşlemcinin uyanması bu kipte 6 saat çevriminden sonra olmaktadır [42,49]. Enerji tüketimini azaltmak için aşağıdaki işlemlerin değerlendirilerek kullanılması gerekmektedir.

- ADC eğer açıksa tüm kiplerde açıktır. Gücü koruyabilmek için uyku kipine geçilmeden ADC kapatılmalıdır.
- IDLE veya ADC NOISE REDUCTION kipe geçerken, Analog karşılaştırıcı eğer kullanılmıyorsa kapatılmalıdır. Diğer uyku kiplerinde Analog karşılaştırıcı zaten otomatik kapatılmaktadır. Fakat Analog Karşılaştırıcısı İç Gerilim İlişkisi için girdi olarak kullanılmaktadır. Bu yüzden bütün uyku kiplerinde kapatılmalıdır.

- Voltaj-Düşüklüğü Algılayıcısı modülü eğer ihtiyaç yoksa kapatılmalıdır. Eğer ki Voltaj-Düşüklüğü Algılayıcısı BODEN FUSE tarafından kullanılıyorsa, tüm uyku kiplerinde açık olmalıdır.
- İç gerilim ilişkisi Voltaj-Düşüklüğü Algılayıcısı, Analog Karşılaştırıcı veya ADC tarafından ihtiyaç duyulursa aktif olur. Eğer bu modüller kapalıysa bu modül kapatılmalı ve enerji tüketmemelidir.
- Gözcü Saati modülüne ihtiyaç yoksa kapatılmalıdır. Eğer açıksa tüm uyku kipleri için açık olmaktadır.
- Uyku kiplerine geçilirken bütün port pinleri minimum güç tüketecek şekilde ayarlanmalıdır. Tüm uyku kiplerinde I/O saati ve ADC saati durdurulursa, cihazın girişleri kapalı olmalıdır. Buda bize ihtiyaç yokken giriş tarafından enerji tüketilmediğini garantiler. Bazı durumlarda giriş için uyanma şartlarını yakalamak gerekmektedir bu durumda açık olabilmektedir [42,51].

Düğümü uyku durumuna geçirme işlemi için güç yönetimi uygulaması olan HPLPowerManagement. Enable() komutunu kendisinin StdControl.Init() metodunda çağırması gerekmektedir ve aşağıdaki durumlardan emin olunmalıdır.

- Radyonun kapandığından (CC1000RadioC. StdControl. stop() komutunun çağrılması gerekmektedir.), Bütün yüksek hızlı kesmelerin kapatıldığından, SPI kesmesinin kapalı olduğundan ve Task kuyruğunun boş olduğundan emin olunması gerekmektedir.
- Bunlardan sonra düğüm bir sonraki saat darbesine kadar uyku kipine geçer. Radyoya ek olarak her hizmette kapatılır. Stop() fonksiyonu içinde, her hizmetin HPLPowerManagement.adjustPower() komutunu çağırması gerekmektedir. Uygulamaların HPLPowerManagement.adjustPower() komutunu çağırmaya ihtiyacı yoktur [45,52].

Uyanma işlemi ise eğer ki düğüm uyku kipine HPLPowerManagement kullanarak geçtiyse, uyanma işlemi bir sonraki 32 kHZ zaman kesmesiyle olmaktadır.

3.2.4. TinyOS 2.X'te güç yönetimi

TinyOS platformu sınırlı enerjiye sahiptir. Tüm cihazlar ve çevre birimleri için birleşik bir güç yönetimi mekanizması mümkün değildir. Örneğin algılayıcı düğümlerin değişik durumlarda önemli ısınma değişiklikleri, güç profilleri ve operasyonel gecikmeleri bulunabilmektedir. Bazı cihazlar örneğin mikro denetleyiciler en düşük güç durumunu çok çabuk hesaplayabilirler. Ancak, algılayıcılar örneğin ısınma zamanlarını hesaplamak için harici bir bilgiye gereksinim duymaktadırlar [53]. Tinyos 2.x güç yönetimi için cihazların 2 tane sınıf tanımlamaktadır. Bunlar işlemciler ve çevre birimleridir. Mikro denetleyiciler birçok güç durumuna sahip olmasına rağmen çevresel birimler 2 farklı duruma sahiptir. Yani ya açıktırlar ya kapalıdırlar. TinyOS'da çevresel birimlerin güç durumlarını yönetmek için 2 tane farklı model bulunmaktadır. Bunlar açık güç yönetimi ve dolaylı güç yönetimi olarak adlandırılırlar [53].

Açık modelde belirlenmiş fiziksel cihazların güç durum kontrolü manuel olarak tek bir istemci için yapılmaktadır. Ne zaman ki bu istemci cihaza güç açık veya güç kapalı durumuna geç derse gecikme olmaksızın bu yapılır(donanımdaki gecikmeler hariç). Bu model özellikle yüksek seviye bileşenlerdeki harici mantıksal birimlerin uygun güç durumunu seçmesi için çok kullanışlı olabilmektedir. StdControl, SplitControl ve AsyncStdControl ara yüzleri bu tipteki güç yönetimi için kullanılabilmektedir [50,53].

Diğer modelde ise cihazın kendisinin kendi sürücüleriyle güç durumunu kontrol etmesine izin verilmektedir. Bu modeldeki cihazların sürücüleri harici istemciler tarafından açılamaz ve durdurulamazlar. Fakat ne zaman güç durumlarının değişeceğine karar verecek bir iç politikaya ihtiyaç duymaktadırlar. Bu politika default olarak cihazın kendi içerisinde bulunabilmekte veya ilk modele benzer şekilde daha alt katmanlarda bu işlem önceki modele benzer şekilde yapılabilmektedir.

Açık güç yönetimi

TinyOS 1.x ile TinyOS 2.x StdControl ve SplitControl ara yüzlerine sahiptirler. Çevre birimlerinin güç durumları için açık güç yönetiminde bu ara yüzler kullanılmaktadır. TinyOS 2.x bunların haricinde üçüncü bir ara yüz olan AsyncStdControl ara yüzünü de kullanmaktadır. Donanımsal bir cihazı, bir bileşenin güç açık veya güç kapalı durumuna geçirebilmesi için bu 3 ara yüzden birinin kullanılması gerekmektedir [51,53].

StdControl ile Güç Yönetimi: Ne zamanki benzetimde güç açık ve güç kapalı durumları arasındaki geçişlerde harcanan zaman ihmal edilebilir ise StdControl arayüzünü kullanmak gerekmektedir. Bu uyanma ve uyuma geçişleri sırasındaki beklemeler bir kaç mikro saniyedir ve ihmal edilebilir. BusyWait ara yüzünde bekleme işlemi yapılabilir. SplitControl ara yüzünde ise bu bekleme zamanı çok daha azdır. Bu yüzden SplitControl ara yüzünü kullanmak gecikmeleri daha da azaltmaktadır. StdControl.start() ve StdControl.stop() ara yüzlerinin açık ve kapalı durumlar için harici olarak çağrılması gerekmektedir. Çağrıldıklarında ise ya FAIL yâ da SUCCESS değerini geri döndürmektedirler. StdControl.start(), bir cihaz için SUCCESS dönerse cihaz tamamen açık duruma gelmektedir. StdControl.stop(), bir cihaz için FAIL geri dönerse cihaz tamamen kapalı duruma gelmektedir. Bu durumda diğer ara yüzler bu cihazın donanımına erişim için çağırım yaptıklarında FAIL veya EOFF değerleri geri dönmektedir. Eğer bir cihaz StdControl.start() veya StdControl.stop() komutlarını talep ettiğinde herhangi bir nedenle bu işlemler tamamlanamazsa geri dönüşüm değeri FAIL olmalıdır. Çizelge 3.2'de stdControl için tanımlı güç durumları gösterilmektedir.

Çizelge 3.2. StdControl bileşeninin geri dönüş değeri[53]

Çağır	**Cihaz Açık**	**Cihaz Kapalı**
StdControl.start()	SUCCESS	SUCCESS, FAIL
StdControl.stop()	SUCCESS, FAIL	SUCCESS
İşletim	Bağlı	FAIL, EOFF

SplitControl ile Güç Yönetimi: Bir cihazın güç kapama ve güç açma durumlarındaki zaman ihmal edilebilir değilse SplitControl ara yüzünün StdControl ara yüzü yerine tercih edilmesi gerekmektedir. Harici bir bileşenin cihazı açmak ve kapamak için SplitControl.start() ve SplitControl.stop() komutlarını çağırması gerekmektedir. Yukarıdaki komutlar çağrılırsa geriye SUCCESS, FAIL, EBUSY, veya EALREADY değerleri dönmektedir [53].

SUCCESS geri dönerse, cihazın çalışmaya başladığı anlamına gelmektedir. Yani güç durumunu değiştirmeye başladığı anlamına gelmektedir ve gelecekte kullanılabilecek bir tamamlama sinyali oluşturulur. EBUSY değeri geri dönerse cihaz başlama ve durma arasındadır (start komutu çağrıldığında başlar veya stop komutu çağrıldığında durmaya başlar) ve bir event için kullanılmaz. EALREADY değeri geri dönerse cihazın hali hazırda aynı durumda bulunduğu anlamına gelmektedir. Hata ve tamamlama sinyali bu durumda üretilmez. FAIL değeri geri dönerse cihazın güç durumunun değiştirilemeyeceği anlamına gelmektedir. Daha açık bir ifadeyle, SplitControl.start() komutu başarıyla çağırılırsa, SplitControl.startDone(SUCCESS) veya SplitControl.startDone(FAIL) şeklinde bir sinyalleşme oluşmakta ve bu aynı zamanda geri dönüşüm değeri olmaktadır. SplitControl.stop() komutu başarıyla çağırılırsa SplitControl.stopDone(SUCCESS) veya SplitControl.stopDone(FAIL) şeklinde bir sinyalleşme oluşmakta ve geri dönüşüm değeri de bu olmaktadır. SplitControl.startDone(SUCCESS) sinyalleşmesi yapılırsa, cihaz tamamen açılmalıdır ve diğer ara yüzler tarafından yapılan işlemsel istekler cihaz tarafından başarıyla sonuçlandırılmalıdır. SplitControl.stopDone(SUCCESS) sinyalleşmesi yapılırsa, cihaz tamamen kapanmalıdır, diğer ara yüzler tarafından yapılan işlemsel istekler yani bu ara yüz tarafından çağrılan komutlar bu cihaz tarafından aynı cihaza erişimin olmadığı yani EOFF veya FAIL değerlerini talepte bulunulan yere dödürmelidir.

Eğer cihaz çalıştıysa ve başarılı bir şekilde SplitControl.stop() komutunu çağırdıysa SplitControl.stopDone(FAIL) değerini sinyaller, cihaz hala tamamen açık olmalıdır ve diğer cihazlardan gelen işlemsel taleplere bu cihaz tarafından başarıyla yanıt

verilebilmelidir [52,53]. Eğerki bir cihaz kapalıysa ve başarılı bir şekilde SplitControl.start() komutunu çağırır ise SplitControl.startDone(FAIL) değerini sinyalleşmektedir ancak cihaz yinede tamamen kapalıdır ve diğer ara yüzlerden gelen işlemsel isteklere bu cihaz tarafından EOFF veya FAIL değerleri geri döndürülebilmektedir. Eğer bir cihaz SplitControl.start() veya SplitControl.stop() komutlarını başarıyla çalıştıramazsa istekler FAIL olarak geri dönmelidir.

Cihaz başlarken SplitControl.start() komutunu veya cihaz kapanırken SplitControl.stop() komutunu çağırdığında geri dönüş değeri SUCCESS olmalıdır. Burada beklenen ise ilgili SplitControl.startDone() veya SplitControl.stopDone() komutlarının ileride sinyalleşmesidir. Cihaz başalarken SplitControl.start() komutu veya cihaz kapanırken SplitControl.stop() komutu çağrıldığında geri dönüş değeri EALREADY olmalıdır, buda cihazın halihazırda o durumda olduğu anlamına gelmektedir. Bu durumda ilgili bitirme olayı (startDone ve stopDone) sinyalleşmemelidir.

Cihaz kapanırken SplitControl.start() komutu veya cihaz başlarken SplitControl.stop() komutu çağrıldığında geri donüş değeri EBUSY olmalıdır, buda cihazın farklı bir işlemle ilgilendiği anlamına gelmektedir. Bu durumda ilgili bitirme olayı (startDone ve stopDone) sinyalleşmemelidir [49]. Çizelge 3.3'te SplitControl bileşeninin geri dönüş değerleri gösterilmektedir.

Çizelge 3.3. SplitControl bileşeninin geri dönüş değeri [53]

Çağır	**Cihaz Açık**	**Cihaz Kapalı**	**Başlarken**	**Kapanırken**
SplitControl.start()	EALREADY	SUCCESS, FAIL	SUCCESS	EBUSY
SplitControl.stop()	SUCCESS, FAIL	EALREADY	EBUSY	SUCCESS
İşletim	Bağlı	FAIL, EOFF, EOFF	FAIL, EOFF, SUCCESS	FAIL, EOFF

Diğer yaklaşımlar SplitControl.start() ve SplitControl.stop() komutlarının geri dönüş değerleri için kabul edilmiştir. Bir diğer yaklaşım ise EBUSY yerine SUCCESS değerinin yer değiştirmesi SplitControl.start() komutunun işlem sonlandırırken ve

SplitControl.stop() komutunun işlem başlatılırken yapılmasına izin vermektedir. Fakat bu yaklaşımın uygulanması cihaz sürücüsünü daha karmaşık hale getirmektedir. EBUSY değerinin geri döndürülmesi en basit yoldur, geri dönüş değeri tam bilinmelidir. Cihaz geçiş durumundaysa EBUSY değerini geri döndürmek, cihaz bileşenlerinin tam olarak start() veya stop() komutlarının neden başarısız olduğunu bilmeleri anlamına gelmektedir ve buna göre aksiyonlarını belirleyebilmektedirler. Cihaz için sadece bir bileşen SplitControl ara yüzünü kullanabilir.

AsyncStdControl ile Güç Yönetimi: StdControl ve SplitControl ara yüzlerinin komut ve olayları eş zamanlıdır ve eş zamanlı olmayan kodlarda çağrılamazlar(örneğin interrupt service routines). Ne zaman bir cihazın güç durumu asenkron kod ile kontrol edilmek istenirse AsyncStdControl ara yüzü kullanılmalıdır.

AsyncStdControl ara yüzü cihaza asenkron içerikte çalışırken güç kapama ve açma izni vermek için kullanılmaktadır. Eğer cihaz asenkron içerikte iken beklenen durma veya açılma olayı gerçekleşmez ise StdControl ara yüzü bu ara yüzün yerine kullanılmaktadır. Eğer bileşenler cihazın kapanması ve açılmasını istiyorlar ise bunu yapmadan önce bir task göndermeleri gerekmektedir. Pratik olarak, AsyncStdControl düşük seviye donanım kaynakları için kullanılmaktadır [52,53].

Dolaylı güç yönetimi

Dolaylı güç yönetimi, güç durumlarının değişimi için bir mekanizma sunmasına rağmen bir politika tanımlamamaktadır. Bu basit durumlarda ve belirlenmiş cihazlar için büyük bir problem ortaya çıkarmamaktadır. Fakat belirlenmemiş durumlarda kritik olmaktadır. Örneğin belirli olmayan durumlarda birçok istemcinin kontrol ettiği cihazlar birbirinden bağımsız ve karmaşık olabilmektedirler.

Örneğin eğer bir bileşen A,B ve C bileşenlerinin istemcisi ise ve A bileşeni stdControl.Stop() komutunu çağırırsa B ve C bileşenlerinin durumu ne olacak? B ve

C bileşenleri durabilecek mi? B ve C kapanırsa A'da kapanabilecek mi? İkisi de uygun ise güç açık ve güç kapalı isteklerine nasıl karar verebilecek? Bu karmaşık yapı TinyOS 1.x'deki stdControl yapısında belirgin sayıda istenmeyen durum ile karşılaşılmasına sebebiyet vermektedir. Birçok platformda, örneğin SPI buses radyo ve flash cihazları arasında paylaşılmaktadır. Tabi ki doğru bir politika SPI bus'ı kullanan istemcileri kontrol eder ve sadece radyo ve flash cihazlarının artık ihtiyacı yok ise kapanmasına izin verir. Tabi ki SPI bus en az bir tane istemci aktif olursa tekrar çalıştırılır [53].

Güç yönetim politikaları

TinyOS 2.x'in sunduğu sistemde genel güç yönetimi politikası gerçek cihazlarda güç yönetimini otomatik yapmaktadırlar. Güç yöneticisi bu politikalardan birisine default olarak sahiptir ve bunu ResourceDefaultOwner ara yüzü ile kullanmaktadırlar.

```
interface ResourceDefaultOwner {
                                async event void granted();
                                async command error_t release();
                                async command bool isOwner();
                                async event void requested();
                                async event void
                                          immediateRequested();}
```

Güç yöneticisi ResourceDefaultOwner.granted() olayından bir sinyali kaynak cihaz üzerinde hakimiyet kurmak için bekler. Bir kere cihaza hâkimiyet kurulursa güç yöneticileri bu politikayı uygulamakta serbesttirler. Eğer uygularlarsa bunu stdControl gibi ara yüzleri kullanarak yaparlar. Farklı güç yöneticileri farklı politikaları kullanabilmektedirler. En basit durum ise acil güç kapama için bir tane stop() komutunu kullanırlar.

Güç yöneticisi güç durumu değişiminde ihmal edilemeyen yani uyanma gecikmesi veya güç tüketimindeki etkileri azaltmak için daha iyi bir strateji uygulayabilir

[50,53]. Güç yönetim politikası ne olursa olsun kullanımı güç yöneticisinde olmaktadır ve kaynak bir istemci tarafından talep edilmediği müddetçe kaynağın sahibi olmaktadır.

Ne zamanki istemci bir talepte bulunursa, güç yöneticisi ResourceDefaultOwner.requested() olayını alır (veya immediateRequested() olayı). Güç yöneticisi bu olaylardan bir sinyal alırsa fiziksel cihazların düşük seviyesinde stdControl gibi ara yüzleri kullanarak kaynağı açar. Bu durumda güç yöneticisi kaynağın sahipliğini bırakmak zorunda kalır. Bu işlemide ResourceDefaultOwner.release() komutunu kullanarak yapmaktadır. Kaynak tamamen açılana kadar güç yöneticisi beklemektedir.

Örnek Güç Yöneticisi PowerManagerC ve DeferredPowerManagerC: TinyOS 2.x'te 2 tane default güç yönetim politikası vardır. Bu politikalar tinyos-2.x/lib/power altında birçok bileşen tarafından uygulanmaktadır. Birinci politika acil güç kontrol şemasını kullanmaktadır. Burada cihazların açılıp kapanması talepte bulunuldukları zaman veya serbest bırakıldıkları zaman acil olarak yapılmaktadır. İkinci politika ise ertelenmiş güç kontrolü şemasını kullanmaktadır. Burada cihaza talepte bulunuluyorsa acil olarak açılır, fakat serbest bırakma olayında küçük bir gecikmeden sonra kapanmaktadır. Bu gecikmede farklı cihaz sürücülerinin değişik ihtiyaçlarının konfigürasyonunun karşılanması durumunda ortaya çıkmaktadır [53].

Her politikanın 3 tane uygulaması bulunmaktadır. Bunlar StdControl, SplitControl, ve AsyncStdControl ara yüzlerinden hangisini kullanacağına göre değişmektedir. Her güç yönetimi için kullanılabilecek bileşenler aşağıda gösterilmiştir.

<u>Acil Güç Yönetimi;</u>
StdControlPowerManagerC
SplitControlPowerManagerC
AsyncStdControlPowerManagerC

Ertelenmiş Güç Yönetimi;

StdControlDeferredPowerManagerC

SplitControlDeferredPowerManagerC

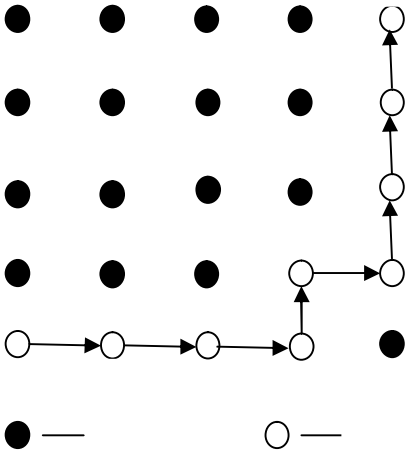
AsyncStdControlDeferredPowerManagerC

4. KABLOSUZ ALGILAYICI AĞLARDA ENERJİ VERİMLİ MAC PROTOKOLÜ: BSC-MAC

BSC-MAC protokolünün hem benzetimi hem de uygulaması gerçekleştirilmiştir. BSC-MAC protokolünün benzetiminden elde edilen sonuçlar P-MAC, S-MAC, AEEMAC protokolleri ile karşılaştırılmıştır. Uygulamada ise BSC-MAC protokolü S-MAC protokolü ile karşılaştırılmıştır.

4.1. BSC-MAC Protokolünün Benzetimi

Şekil 4.1'de P-MAC protokolünün 25 düğümlük topolojisi gösterilmektedir. Aynı topoloji BSC-MAC protokolü içinde gerçekleştirilmiştir. P-MAC protokolünde ns-2 benzetim aracı kullanılmıştır [24].

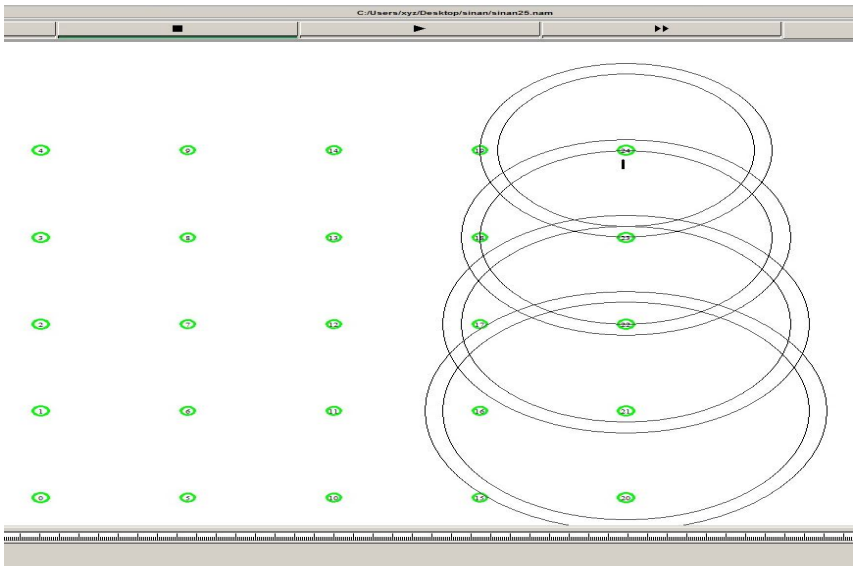


Şekil 4.1. P-MAC örnek topoloji

ns-2 KAAiçin gerekli ortamları sunmaktadır. Şekil 4.2'de ns-2 ortamının 25 düğümlük topolojisi görülmektedir. Topolojinin bu şekilde oluşturulmasının nedeni ise P-MAC ile aynı topolojiye sahip olması sebebiyledir [24]. Sağ üsteki düğüm baz istasyonu olarak çalışmaktadır. İlk önce tüm düğümler iletişim yapmakta ve baz istasyonuna akışlar için tüm yollar oluşturulmaktadır. Bundan sonraki işlem ise hangi

düğümün kök veya kaynak olarak çalışacağının belirlenmesi Şekil 4.3'de akış diyagramında da gösterildiği gibi bulunmakta ve iletişim başlamaktadır.

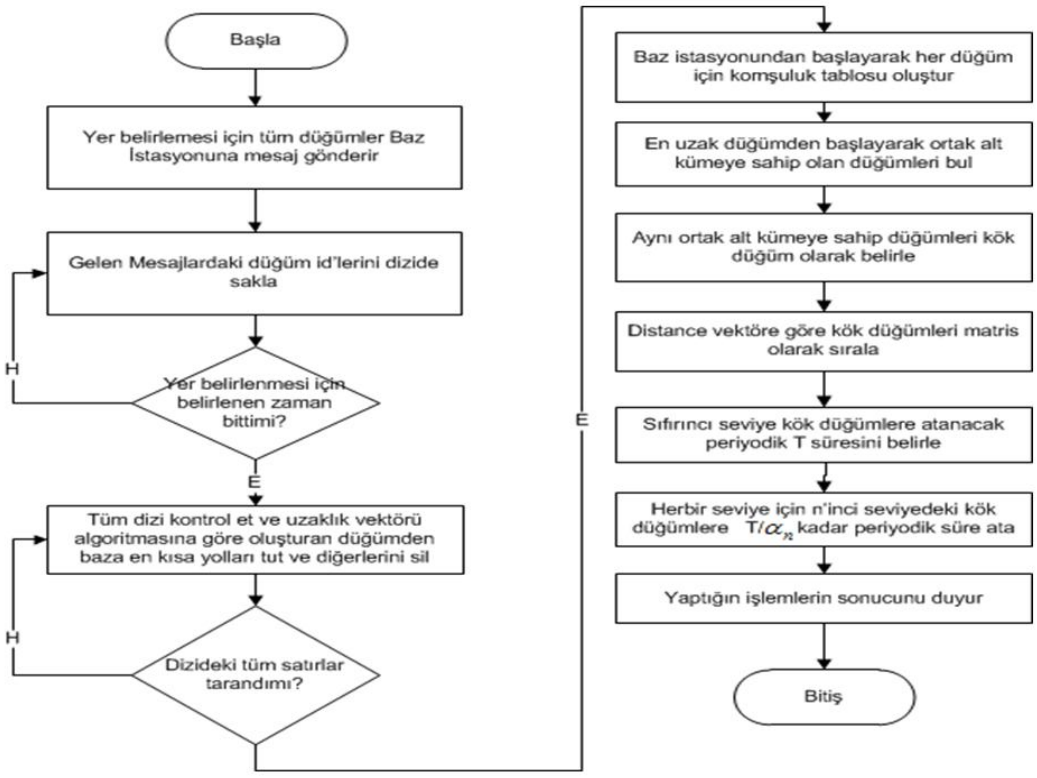
Uyku süreleri ise şu şekilde ayarlanmaktadır. Kaynak düğümler kendi aralarında iletişim yapıp bir birlerini uyku durumuna geçirmektedirler. Örneğin ilk hangi düğüm iletişime başlarsa gönderdiği tüm paketler broadcast yayın yaptığından dolayı tüm komşularını direk uyku moduna geçirmektedir. Belirlenen süre dolduğunda tüm düğümler uyanmakta ve yolu ilk ele geçiren diğerlerine mesaj atmakta ve onları uyku durumuna geçirmektedir. İstem dışı alım diğer protokollere göre çok daha az gerçekleşmektedir. Çünkü ilk mesajı atan diğerleri ayrı bir mesaj atmadan onları direk uyku moduna geçirmektedir.



Şekil 4.2. Gerçekleştirilen benzetimin örnek gösterimi

Kök düğümler içinse durum kaynak düğümlere göre çok daha farklıdır. Burada kök düğümlerin ne zaman uyuyacağına tamamen baz istasyonu karar vermektedir. Buradaki en temel problem ise zaman senkronizasyonudur. Ancak baz istasyonu durdurup uyandırdığı için zaman senkronizasyonu baz istasyonunun zamanına göre ayarlandığından çok fazla bir gecikmeye ve paket kaybına sebep olmamaktadır. Yapılan uygulamada başlangıçta tüm düğümler çalışmakta ancak belirli bir süre sonra kimin ne zaman çalışacağı belli olmakta bu yüzden istem dışı alım azalmakta,

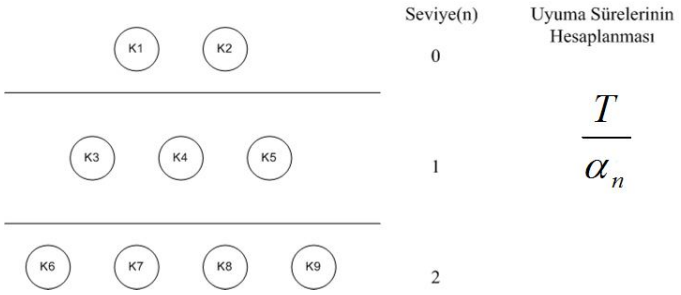
enerji verimliliği arttırılmakta ve devamlı bir iletişim sağlandığından iletişim daha fazla ve adaletli olmaktadır. Bu sayede paket kayıpları azaltılmaktadır. Düğümlerin uyku ve uyanma arasındaki gecikme süreleri de kaynak düğümlerde hemen hemen hiç olmamakta kök düğümlerinde ise çok az olmaktadır.



Şekil 4.3. Düğümlerin kök kaynak durumlarını belirleyen akış diyagramı

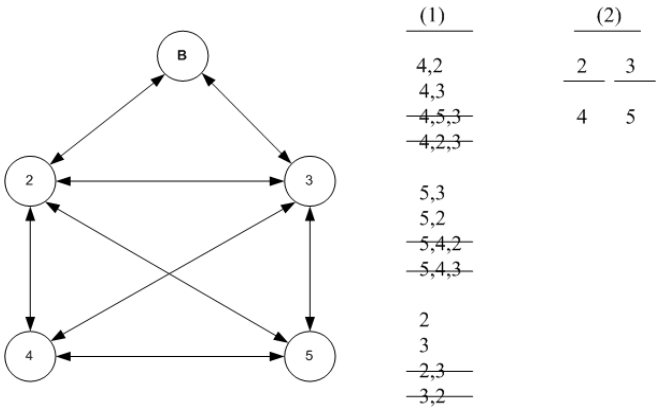
Şekil 4.3'de gösterilen akış diyagramı baz istasyonunda çalışan ve hangi düğümlerin kök düğüm olduğuna karar veren bir yapıdır. Bunun için ilk önce tüm düğümlerin listesi baz istasyonuna gelmektedir. Buda flooding şeklinde yapılmakta ve belirli bir süre için buna izin verilmektedir. Bu süre topolojiye göre değişmektedir. Zamanın hesaplanması bir paketin bir düğümden geçişi için gereken zaman ile bir paketin baz istasyonuna ulaştırılması için gereken maksimum adım sayısı ve geri çekilmeden dolayı olabilecek maksimum gecikmenin çarpımı ile bulunmaktadır. Daha sonra oluşturulan dizilerdeki yollar uzaklık vektörü algoritmasına göre en kısa yol olacak şekilde ayarlanmaktadır. Diğer yollar ise diziden atılmaktadır. Baz istasyonundan başlayarak komşuluk tablosu oluşturulmakta ve sonra en uzak noktadaki düğümden

başlanarak ortak alt kümeye sahip düğümler bulunmaktadır. Aynı ortak alt kümeye sahip olan düğümler kök düğüm olarak belirlenmekte ve uzaklık vektörü algoritmasına göre elde olan kök düğümler sıralanır ve bunlar seviye seviye ayrılmaktadır. Her seviye için bir zaman değeri belirlenmekte ve bu sayede hangi kök düğümlerin hangi zaman aralıklarında uyuyacağı ayarlanmaktadır. Şekil 4.4'de Şekil 4.3'de gösterilen akış diyagramındaki kök düğümlerinin uyuma zamanlarının belirlenmesi işlemi gösterilmektedir.



Şekil 4.4. Kök düğümlerinin uyuma sürelerinin hesaplanması

Şekil 4.4'te gösterildiği gibi baz istasyonuna en yakın olan sıfırıncı seviyedir. Dolayısıyla en yakında olan gelen paketi baz istasyonuna bir adımda gönderebilmektedir. Dolayısıyla buradaki kök düğüm sayısı az olacağından uyuma süresi adaletli erişim mantığına göre yarı yarıya olmaktadır. Dolayısıyla bu uyuma zamanlarını formülize etmek gerekirse her seviyedeki kök düğümlerin uyuma zamanları T/α_n olarak belirlenmektedir. Bu sayede mesajı oluşturan kaynak düğümün baz istasyonuna gönderdiği mesajların kaybolması önlenmekte ve muhakkak bir yol bulması sağlanmaktadır. Şekil 4.5'te ise akış diyagramındaki kök düğümlerin bulunması işlemi bir örnek üzerinden gösterilmektedir.



Şekil 4.5. Kök düğümlerin belirlenmesi işleminin örnek gösterimi

Şekil 4.5'te gösterildiği gibi 4 tane düğüm ve bir baz istasyonu olduğu varsayılırsa tüm düğümler başlangıçta flooding yapmaktadırlar. Bu sayede tüm düğümlerin id'leri baz istasyonuna gelmektedir. Gelen düğümlerin hangi yolu izledikleri (1) numaralı kısımda gösterilmektedir. Uzaklık vektörü algoritmasına göre en kısa yol belirlenmektedir ve diğer düğümler silinmektedir. Daha sonra baz istasyonundan başlayarak komşuluk tablosu oluşturulmaktadır ve sonra en uzaktaki düğümden başlanarak ortak alt kümeye sahip düğümler belirlenmektedir. Bu işlem (2) de gösterilmektedir ve bu ortak alt kümeye sahip düğümler kök düğüm olarak belirlenmektedir. Bu mantık sayesinde ve düğümlerin uyutulup uyanması ile hem adaletli bir iletişim hem iyi derecede bir güç verimliliği elde edilmektedir.

Çizelge 4.1. Benzetimde kullanılan değerler [24]

Parametre	**Değer**
Başlangıç Enerjisi	100 Joules
İletim Gücü	0,5 Watts
Alım Gücü	0,3 Watts
Boş Güç	0,05 Watts
Bant Genişliği	0,5 Kbps
Veri cwnd	63 ms

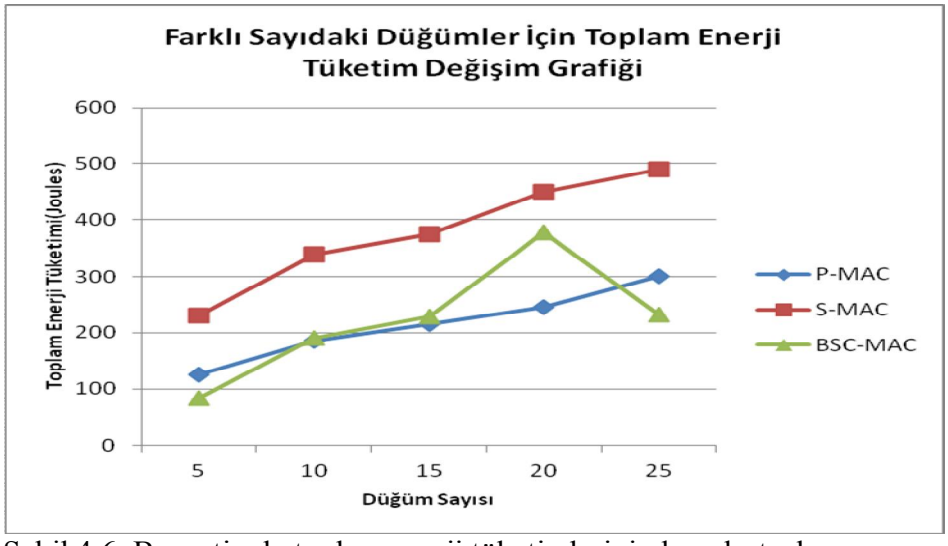
Elde edilen sonuçlarda P-MAC ve S-MAC için P-MAC makalesinden yararlanılmıştır [24]. Çizelge 4.1'de benzetim esnasında kullanılan değerler gösterilmektedir. BSC-MAC için ise Çizelge 4.1'deki değerler kullanılarak ns-2 benzetim ortamında önerilen kodun gerçekleştirimi yapılmıştır. Her bir benzetim 10

kere çalıştırılarak alınan sonuçların ortalaması buraya eklenmiştir. Benzetim için 5, 10, 15, 20 ve 25 düğümlü topolojiler kullanılarak 5 farklı senaryo üzerinden benzetimleri gerçekleştirilmiştir.

Çizelge 4.2. Toplam enerji tüketimi değerleri

Düğüm Sayısı / Protokol	5	10	15	20	25
P-MAC(joule)	125	185	215	245	300
S-MAC(joule)	230	340	375	450	490
BSC-MAC(joule)	83	190	229	378	232

Çizelge 4.2'de P-MAC, S-MAC ve BSC-MAC için 1500 sn yapılan benzetim için ve bütün topolojiler için elde edilen toplam enerji tüketimleri gösterilmektedir. Elde edilen sonuçlarda BSC-MAC yönteminin S-MAC yönteminden daha iyi olduğu ancak topoloji farklılıklarından dolayı toplam enerji tüketiminim P-MAC yönteminden bazı topolojilerde biraz daha fazla olduğu görülmektedir. Çizelge 4.2'de elde edilen sonuçların grafiksel gösterimi Şekil 4.6'da gösterilmektedir.



Şekil 4.6. Benzetimde toplam enerji tüketimlerinin karşılaştırılması

Şekil 4.6'da yeşil renk ile gösterilen BSC-MAC yönteminin benzetim sonucunda elde edilen ortalama grafiğini göstermektedir. 5 ve 25 düğümlü topolojilerde BSC-

MAC yönteminin P-MAC ve S-MAC yönteminden daha iyi olduğu ortaya çıkmaktadır. 10 ve 15 düğümlü topolojilerde ise elde edilen sonuçların S-MAC yönteminden daha iyi olduğunu ancak P-MAC yöntemiyle hemen hemen benzer bir enerji tüketimi olduğu görülmektedir. 20 düğümlü topolojide ise BSC-MAC yönteminin S-MAC'ten daha iyi, P-MAC protokolünden daha kötü çıktığı görülmektedir.

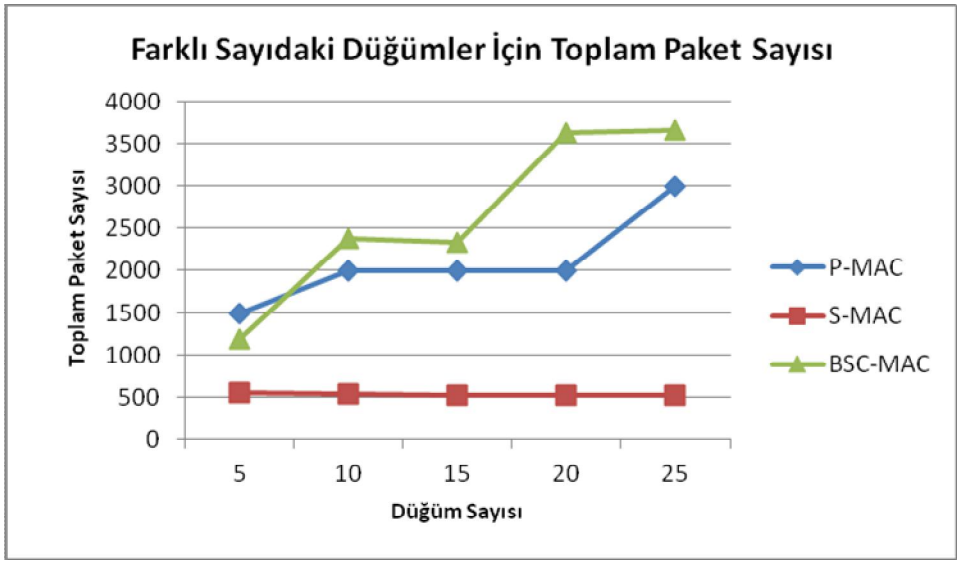
BSC-MAC ve diğer protokollerin bir diğer karşılaştırılması 1500 saniye için yukarıda bahsedilen tüm topolojilerde toplam alınan paket sayısı olarak gerçekleştirilmiştir. Çizelge 4.3'te ns-2 benzetim ortamında yapılan toplam iletilen paket sayısının benzetimin sonuçları gösterilmektedir.

Çizelge 4.3. Toplam iletilen paket sayısı değerleri

Düğüm Sayısı / Protokol	**5**	**10**	**15**	**20**	**25**
P-MAC	1500	2000	2000	2000	3000
S-MAC	550	540	530	530	530
BSC-MAC	1195	2380	2331	3640	3658

Çizelge 4.3'te gösterildiği gibi 5 düğümlü topoloji hariç tüm diğer düğümlerde alınan toplam paket sayısı BSC-MAC yönteminde diğer protokollere göre çok daha iyi sonuç vermektedir. Burada bir önceki şekilde BSC-MAC yöntemi için çıkan toplam enerji tüketiminin P-MAC yöntemine göre daha kötü çıkmasının en büyük nedeni topolojide oluşturulan toplam paket sayısından kaynaklanmaktadır [24]. Örneğin BSC-MAC protokolü P-MAC protokolünden 5 düğümlü topolojide enerji tüketimi bakımından daha iyi çıkarken üretilen toplam paket bakımından daha kötü çıkmaktadır. Topolojinin büyümesiyle birlikte üretilen paket sayısı BSC-MAC yönteminde çok daha fazla olduğu için enerji tüketiminin Şekil 4.5'teki gibi biraz fazla çıkması gayet normal bir durumdur. 20 düğümlü topolojide ise toplam üretilen paket sayısı diğer protokollerden çok daha fazla olduğu için burada tüketilen enerjide daha fazla olduğu görülmektedir. Şekil 4.7'de ise P-MAC, S-MAC ve BSC-MAC

için benzetim sonucunda elde edilen toplam paket sayıları grafiksel olarak karşılaştırmalı bir biçimde gösterilmektedir.



Şekil 4.7. Benzetimde iletilen toplam paket sayılarının karşılaştırılması

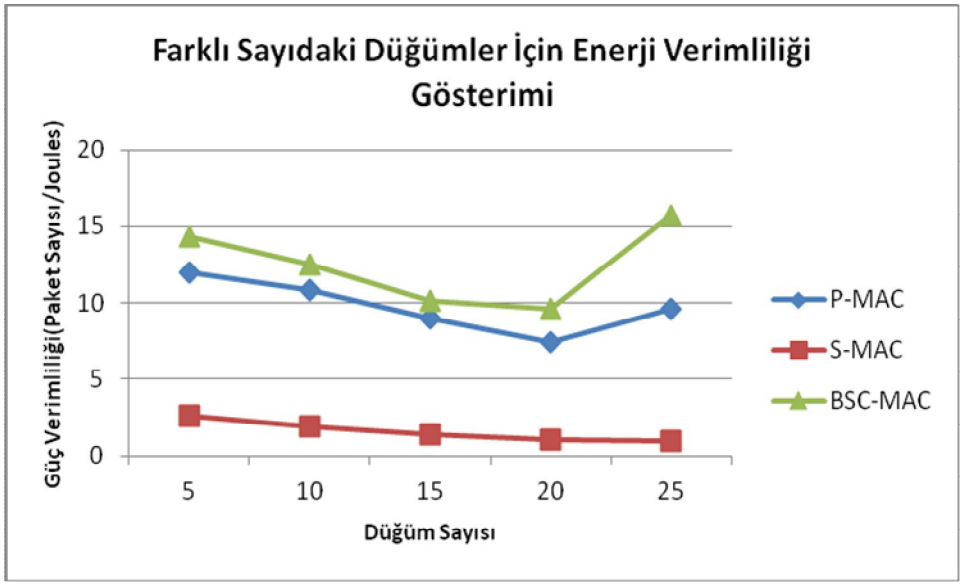
Şekil 4.7'den de anlaşılacağı gibi BSC-MAC yönteminin değişik topolojilerde 1500 sn yapılan benzetim için elde edilen sonuçlarda diğerlerine göre çok daha iyi sonuç verdiği görülmektedir. 5 düğümlü topolojide P-MAC yönteminden daha az çıkmasının nedeni ise uyuma ve uyanma algoritmasının 5 düğüm gibi az bir topolojide istenen verime ulaşmaması şeklinde yorumlanması gerekmektedir. Büyük topolojilerde BSC-MAC protokolünün diğerlerine göre daha iyi sonuçlar vereceği elde edilen sonuçlardan anlaşılmaktadır.

Benzetim ortamında karşılaştırılan sonuçlarda daha kesin sonuç elde edebilmek için topolojiler bazında alınan paket sayısı ile topolojiler bazında tüketilen toplam enerji miktarları bir birlerine oranlanmıştır. Elde edilen sonuçlar Çizelge 4.4'te gösterilmektedir.

Çizelge 4.4. Güç verimliliği değerleri (paket sayısı/ joules)

Düğüm Sayısı / Protokol	5	10	15	20	25
P-MAC	12	10,8	8,98	7,4	9,6
S-MAC	2,6	1,98	1,4	1,1	1
BSC-MAC	14,3	12,5	10,17	9,62	15,76

Çizelge 4.4'e bakıldığında üretilen paket sayısına göre tüketilen enerji oranları görülmekte ve BSC-MAC yönteminin diğer yöntemlere göre verimlilik açısından çok daha iyi olduğu görülmektedir. Topolojide bulunan toplam düğüm sayısı arttıkça BSC-MAC protokolü daha iyi sonuç vermektedir. Şekil 5.8'de protokollerin topoloji bazında enerji verimlilikleri gösterilmektedir.



Şekil 4.8. Benzetimde güç verimliliklerinin karşılaştırılması

Şekil 4.8'de gösterilen protokollerin karşılaştırmasında enerji verimliliği açısından elde edilen sonuçların BSC-MAC protokolü için diğer protokollere göre daha başarılı sonuç verdiği görülmektedir. Dolayısıyla bir ağın enerji verimliliği sadece tükettiği enerji değil yaptığı işe ve ortaya çıkan verime göre değişmektedir. BSC-MAC protokolünde üretilen paket sayısı az olursa enerji tüketiminin diğer protokollere

göre daha az olacağı da bu elde edilen sonuçlardan anlaşılmaktadır. Çizelge 4.5'te karşılaştırmada kullanılan tüm protokollerin 2000 paket üretmeleri halinde tüketecekleri toplam enerji miktarları gösterilmektedir.

Çizelge 4.5. Aynı sayıda paket gönderiminde enerji

Düğüm Sayısı / Protokol	5	10	15	20	25
P-MAC	166,66	185	215	245	200
S-MAC	836,36	1259,25	1415,09	1698,11	1849,05
BSC-MAC	138,91	159,66	196,48	207,69	126,84

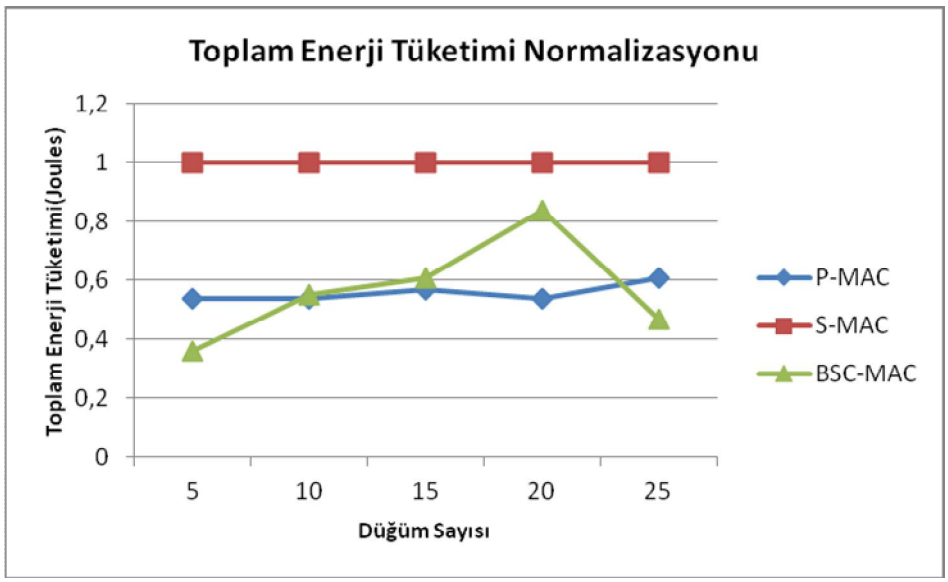
Çizelge 4.5'te gösterildiği gibi eğer baz istasyonuna ulaşan paketlerin 2000 olması durumunda tüketilen enerji miktarları gösterilmekte ve Çizelge 4.5'ten de anlaşılacağı gibi BSC-MAC protokolleri diğer protokollerden daha az enerji tüketerek 2000 paketin gönderilme işlemini gerçekleştirdiği görülmektedir.

Çizelge 4.6'da toplam enerji tüketimlerinin normalizasyon değerleri gösterilmektedir. Her kolon için en yüksek değer 1 olarak alınmış ve diğer değerlerin bu değere oranı alınarak oransal bir gösterim elde edilmiştir.

Çizelge 4.6. Toplam enerji tüketimi normalizasyon değerleri (paket sayısı/ joules)

Düğüm Sayısı / Protokol	5	10	15	20	25
P-MAC	0,54	0,54	0,57	0,54	0,61
S-MAC	1	1	1	1	1
BSC-MAC	0,36	0,55	0,61	0,84	0,47

Çizelge 4.6'da görüldüğü gibi enerji tüketimi olarak en fazla tüketim S-MAC yönteminde ortaya çıkmaktadır. Diğerlerinin S-MAC yöntemine oranına bakıldığında bazı topolojilerde P-MAC bazı topolojilerde ise BSC-MAC daha iyi sonuç vermektedir. Şekil 4.9'da ise tablodan elde edilen sonuçlar grafiksel olarak gösterilmektedir.



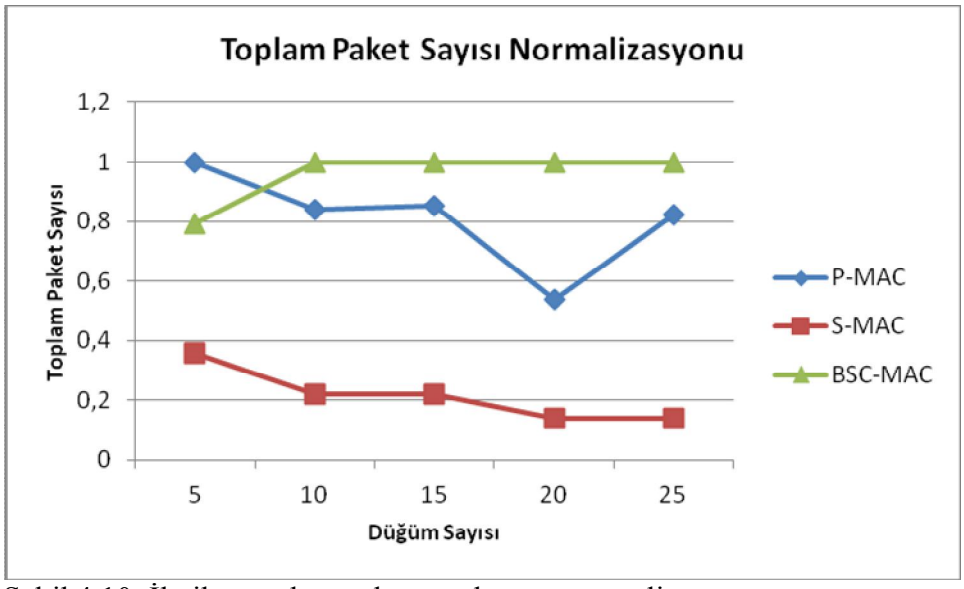
Şekil 4.9. Toplam enerji tüketimlerinin normalizasyonu

Şekil 4.9'a baktığımızda S-MAC yöntemi 1500 saniyelik benzetim sonucunda en fazla enerji tüketen yöntem olarak görmekteyiz. P-MAC ve BSC-MAC yöntemlerinin ise enerji tüketimleri topolojilere göre farklılıklar göstermektedir. Çizelge 4.7'de topolojiler bazında ağda baz istasyonu tarafından alınan toplam paket sayıları oransal olarak gösterilmektedir.

Çizelge 4.7. Toplam iletilen paket sayısı normalizasyon değerleri

Düğüm Sayısı / Protokol	**5**	**10**	**15**	**20**	**25**
P-MAC	1	0,84	0,85	0,54	0,82
S-MAC	0,36	0,22	0,22	0,14	0,14
BSC-MAC	0,79	1	1	1	1

BSC-MAC protokolünün diğer protokollere göre iletişim konusunda daha başarılı olduğu görülmektedir. Özellikle 20 düğümlü topolojide ki oran diğerlerinin çok üstündedir. Bundan dolayı enerji tüketimi bir önceki grafikte P-MAC yönteminden daha kötü çıkmıştır. Şekil 4.10'da 1500 saniyelik benzetim sonucunda elde edilen değerlerin normalizasyon grafiği gösterilmektedir.



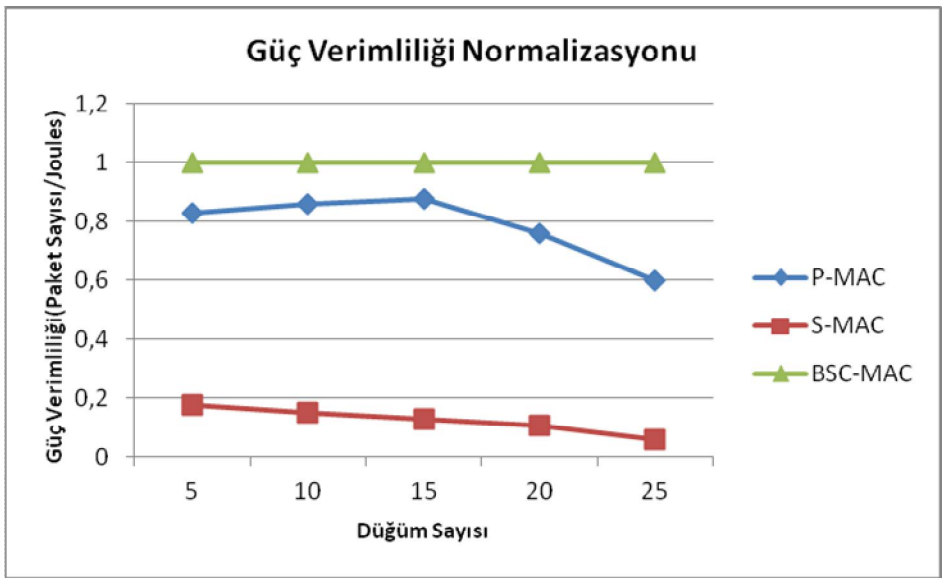
Şekil 4.10. İletilen toplam paket sayılarının normalizasyonu

Şekil 4.10'da görüldüğü gibi BSC-MAC paket iletişimi konusunda diğer yöntemlerden daha iyi sonuç vermektedir. Özellikle 20 düğümlük topolojide bu daha iyi anlaşılmaktadır. Çizelge 4.8'de ise farklı topolojilerdeki enerji verimliliği için normalizasyon işlemini görmekteyiz.

Çizelge 4.8. Güç verimliliği normalizasyon değerleri

Düğüm Sayısı / Protokol	**5**	**10**	**15**	**20**	**25**
P-MAC	0,83	0,86	0,88	0,76	0,6
S-MAC	0,18	0,15	0,13	0,11	0,06
BSC-MAC	1	1	1	1	1

Çizelge 5.8'de görüldüğü gibi bir ağ için çok önemli olan enerji verimliliği için benzetim sonucunda elde edilen sonuçların normalizasyon işlemi sonucu elde edilen sonuçlar görülmektedir. Enerji verimliliği açısından baktığımızda BSC-MAC yönteminin diğer yöntemlere nazaran 5 farklı topolojide de daha iyi sonuç verdiği oransal olarak görülmektedir. Şekil 4.11'de ise enerji verimliliği normalizasyon işlemi grafiksel olarak gösterilmektedir.



Şekil 4.11. Güç verimliliklerinin normalizasyonu

Şekil 4.11'de görüldüğü gibi enerji verimliliği açısından BSC-MAC diğer yöntemlere nazaran daha iyi sonuç vermektedir. Özellikle topolojideki düğüm sayısı arttıkça sonuçlar daha iyi çıkmaktadır.

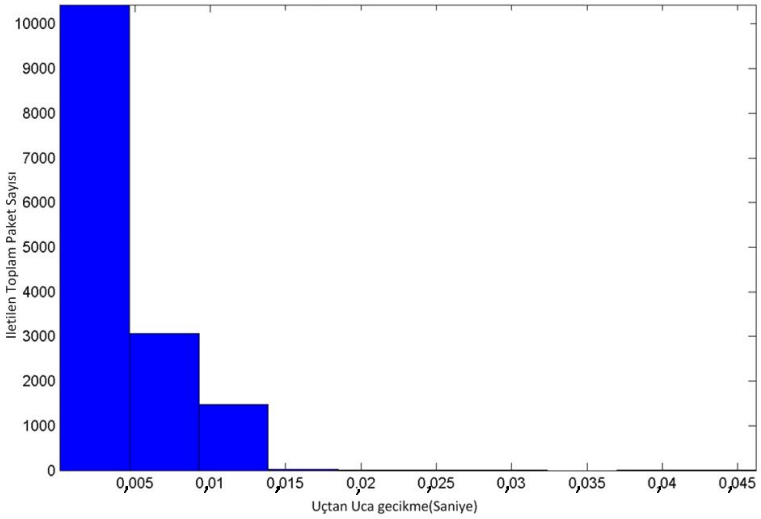
BSC-MAC protokolü P-MAC protokolünün haricinde bir de AEEMAC protokolü ile karşılaştırılmıştır[33]. AEEMAC protokolü geliştirdiği protokolü S-MAC protokolü ile karşılaştırmıştır. Bunun için ns-2 benzetim aracını kullanılmıştır. Karşılaştırma işlemi için 2 topoloji kullanılmıştır. Bunlar sırasıyla doğrusal ve ızgara topolojileri şeklinde adlandırılmıştır.

Çizelge 4.9'da AEEMAC ve BSC-MAC protokollerinin karşılaştırılması için kullanılan parametreler ve değerler gösterilmektedir. Bu değerler AEEMAC protokolünde kullanılmıştır. Karşılaştırmanın doğru bir şekilde yapılması için BSC-MAC benzetiminde de aynı değerler kullanılmıştır [34].

Çizelge 4.9. AEEMAC benzetiminde kullanılan parametre ve değerler [34]

Parametre	**Değer**
Radyo Yayılım Modeli	TwoRayGround
Yönlendirme Protokolü	DSDV
Başlangıç Enerjisi(Joule)	1500
Alım Gücü(Joule)	1,4
İletim Gücü(Joule)	1,7
Güç Geçişi(Joule)	0,055
Boş Güç & Uyku Gücü (Joule)	1,0 & 0,002
Geçiş Süresi(Saniye)	0,015
Toplam Benzetim Süresi(Saniye)	1400
Tarfik Oranı(Kbps) / Paket Boyutu(B)	512 / 512
Trafik Türü	CBR

Şekil 4.12'de doğrusal topolojide BSC-MAC protokolünün benzetimi sonucunda uçtan uca gecikme için elde edilen sonuçlar gösterilmektedir. ns-2 benzetim aracında AEEMAC protokolü ile aynı koşullarda çalıştırılan BSC-MAC protokolü için elde edilen uçtan uca gecikme grafiği kullanılarak benzetim esnasındaki gecikmeler direk elde edilmiştir [34].



Şekil 4.12. Benzetim sonucunda uçtan uca gecikme için elde edilen sonuçlar

Şekil 4.12'de gösterilen değerler kullanılarak elde edilen sonuçlar Çizelge 4.10'da doğrusal topolojide elde edilen gecikme olarak gösterilmektedir. Toplam üretilen

paket sayısına göre elde edilen gecikme süresi oranlanmış ve bu sayede o paketler için toplam gecikme süresi elde edilmiştir.

Çizelge 4.10. Doğrusal topoloji benzetim sonucu elde edilen toplam gecikme

Toplam Paket Sayısı	**Gecikme Süresi(sn)**	**Toplam Gecikme(sn)**
10431	0,005	52,15
3070	0,01	30,7
1469	0,0139	20,41
23	0,15	0,34
10	0,023	0,23
7	0,025	0,17
Paket Gönderim ve Alım Esnasındaki Toplam Gecikme		104,04

Yani 10431 paket için 0,005 sn gecikme elde edilmiştir. Bunların birbirleri ile oranlanması ile üretilen 10431 paket için toplam 52,15 saniyelik bir gecikme elde edilmiştir. Diğer paketler içinde aynı işlemler yapılmış ve elde edilen değerler Çizelge 4.10'da gösterilmiştir. AAEMAC protokolünün benzetim için kullandığı parametreler kullanıldığında elde edilen toplam sonuç 104,04 saniyedir. Yani benzetim esnasında elde edilen uçtan uca toplam gecikme 104,04 saniye olarak hesaplanmıştır.

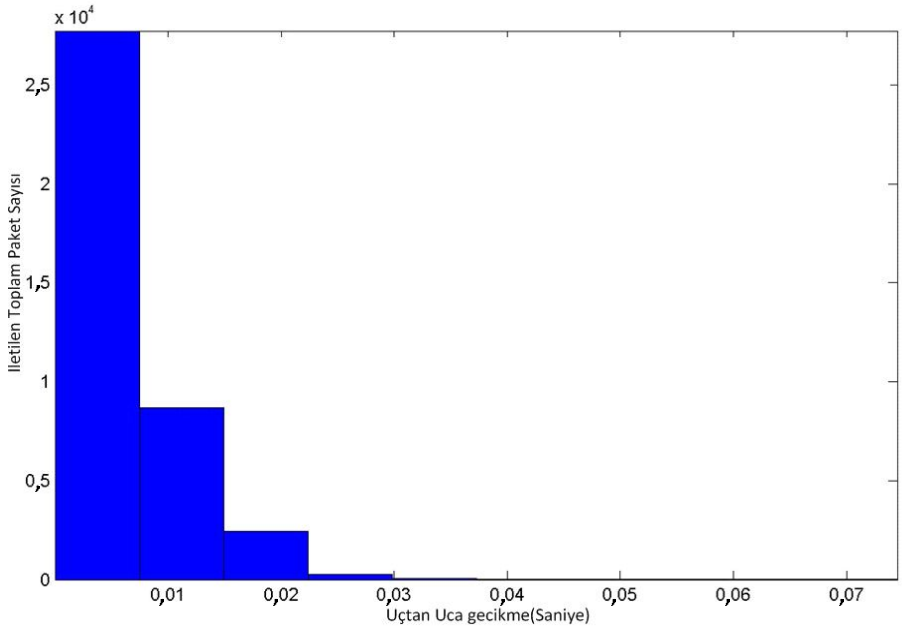
Kanal Gecikmesi: Bir iletişim sırasında elde edilen benzetim gecikmesinin haricinde birde kanal gecikmesi bulunmaktadır. Bundan dolayı tam bir karşılaştırma için kanalda meydana gelecek gecikmeler şu şekilde hesaplanmıştır.

- 1 Paket=100 Byte x 8=800 bit transfer olmaktadır.
- Kanal=1 Mbps olarak elde edilmiştir.
- 1 Paketin kanaldan geçiş süresi=800 / 1000000=0,8 ms olarak elde edilmiştir.
- Düğüm başı üretilip karşı tarafın aldığı paket =0,8ms x 3=2,4 ms olarak elde edilmiştir.
- Dolayısıyla toplamda 9,072 saniye kanal gecikmesi elde edilmektedir.

Geri Çekilmeden Dolayı Meydana Gelen Gecikme: 802.11'e göre her paket için ortalama 140 mikro saniye geri çekilme gecikmesi bulunmaktadır. Dolayısıyla Toplam geri çekilme gecikmesi benzetim esnasında üretilen toplam paket sayısı ile geri çekilme gecikmesinin çarpılması sonucu elde edilebilmektedir. Dolayısıyla toplam geri çekilme gecikmesi 15,011x140=2,11 sn olarak elde edilmektedir.

Dolayısıyla uçtan uca iletişimde toplam gecikme benzetimde elde edilen toplam gecikme ile kanal gecikmesinin ve geri çekilmeden dolayı ortaya çıkan gecikmenin toplanması sonucu elde edilmektedir. Dolayısıyla elde edilen toplam gecikme 104,04+9,072+2,11=115,22 saniye olarak elde edilmektedir.

Şekil 4.13'de ızgara topolojinin benzetiminde uçtan uca gecikme için elde edilen sonuçlar gösterilmektedir. ns-2 benzetim aracında AEEMAC protokolü ile aynı koşullarda çalıştırılan BSC-MAC protokolü için elde edilen uçtan uca gecikme grafiği kullanılarak benzetim esnasındaki gecikmeler direk elde edilmiştir.



Şekil 4.13. Izgara topolojinin benzetiminde uçtan uca gecikme

Şekil 4.13'de gösterilen değerler kullanılarak elde edilen sonuçların birbirlerine oranlanması ile ızgara topoloji için toplam gecikme süresi hesaplanmıştır. Çizelge 4.11'da benzetim sonucu elde edilen toplam gecikme ve bunların toplamı gösterilmektedir.

Çizelge 4.11. Izgara topoloji benzetim sonucu elde edilen toplam gecikme

Toplam Paket Sayısı	**Gecikme Süresi(sn)**	**Toplam Gecikme(sn)**
27731	0,0074	194,117
8707	0,0149	129,73
2453	0,0224	54,94
262	0,029	7,80
66	0,037	2,44
25	0,045	1,125
20	0,052	1,04
4	0,052	0,20
Paket Gönderim ve Alım Esnasındaki Toplam Gecikme		391,392

Kanal gecikmesi=27,972 saniye olarak elde edilmiştir.
Geri çekilme gecikmesi= 4,8951 saniye olarak elde edilmiştir.

Dolayısıyla ızgara topolojide uçtan uca iletişimde toplam gecikme 391,392+ 27,972+4,8951=424,2591 saniye olarak elde edilmektedir. Belirlenen topolojilerin her biri için toplam kalan enerji miktarı, uçtan uca gecikme, paket ulaştırma oranı ve üretilen iş parametreleri üzerinden karşılaştırma yapılmıştır. Benzetim sonucunda elde edilen sonuçlar Çizelge 4.12'de gösterilmektedir [34].

Çizelge 4.12. Benzetimde elde edilen sonuçlar

Protokoller / **Parametreler** / **Topoloji**		**S-MAC**	**AEEMAC**	**BSC-MAC**
Toplam Kalan Enerji (J)	Doğrusal	1273,25	1285,33	1422,53
	Izgara	957,37	1289,94	1286,26
Uçtan Uca Gecikme (s)	Doğrusal	139,728	140,188	115,22
	Izgara	689,67	464,438	424,259
Paket Ulaştırma Oranı (%)	Doğrusal	39,8	39,2	39,9
	Izgara	6,236	14,719	23,07
Üretilen İş (Kbps)	Doğrusal	1,89	1,87	1,91
	Izgara	0,39	1,09	1,82

BSC-MAC protokolü ile AEEMAC protokolünün karşılaştırması için ise doğrusal ve ızgara topolojileri kullanılmıştır. Çizelge 4.12'de gösterildiği gibi BSC-MAC protokolü gerçekleştirilen benzetim sonucunda AEEMAC ve S-MAC protokollerinden karşılaştırma parametreleri için ayrı ayrı denenmiş ve BSC-MAC protokolünün diğer protokollerden daha iyi sonuç verdiği elde edilen sonuçlardan anlaşılmıştır.

Bant genişliğini hangi protokolün daha iyi kullandığını anlayabilmek için 10 kbit Bant Genişliği ve 10 kbit paket boyutundaki trafik yoğunluğunda gerçekleştirilen benzetim sonucunda elde edilen değerler aşağıda gösterilmektedir. Çizelge 4.13'de S-MAC ve BSC-MAC protokollerinin bant genişliğinden yararlanma miktarları için karşılaştırmaları gösterilmiştir [22].

BSC-MAC; Gönderilen = 7528
Alınan = 1144
} %15,20

S-MAC; Gönderilen = 6422
Alınan = 962
} %14,98

Çizelge 4.13. Benzetimde elde edilen bant genişliğinden yararlanma değerleri

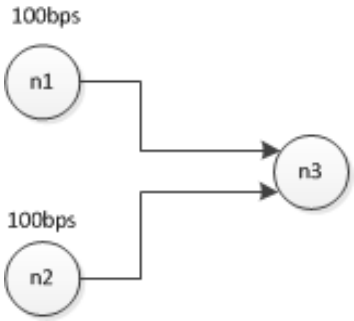
Protokol / Parametre	S-MAC	BSC-MAC
Bant Genişliğinden Yararlanma(%)	14,98	15,2

Gerçekleştirilen benzetimde bant genişliği 10 Kbps olarak alınmıştır. S-MAC ve BSC-MAC protokolleri için paket boyutu da 10 Kbit olarak ayarlanmıştır. Benzetim sonucunda elde edilen değerler Çizelge 4.13'de gösterilmektedir. Dolayısıyla BSC-MAC protokolünün bant genişliğini S-MAC protokolünden daha iyi şekilde kullanıldığı görülmektedir [22].

Yukarıdaki karşılaştırma parametreleri haricinde birde bu protokollerin karşılaştırılmaları için Call Blocking parametresi temel alınmıştır. Bu karşılaştırma analitik olarak yapılmış ve Erlang yöntemi kullanılmıştır. Erlang(E) telefon santrallerinde, servis sağlayıcı elementlerin üzerinde telefon devresi veya telefon anahtar cihazları gibi gelen yük veya taşınan yükün ölçümü olarak kullanılan boyutsuz bir birimdir [54]. Oluşan trafiğin bir ölçüm şeklide anlık trafik içindir. Erlang formülü Eşitlik 1'de gösterilmektedir.

$$E = \lambda h \qquad \text{(Eş.1)}$$

λ gelen paket sayısı, h ise bir paketin bir düğümden geçmesi için gereken zamandır [54]. Gerçekleştirilen benzetimde her paket için 100 byte/saniye kullanılmıştır. Ayrıca bant genişliği ise 10kbps olarak değiştirilmiştir. Dolayısıyla Şekil 4.14'de görüldüğü gibi bir düğüme 2 düğüm paket yollayabilmektedir.



Şekil 4.14. Örnek trafik modeli

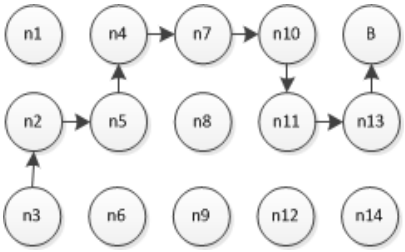
Şekil 4.14'de n3 düğümüne n1 ve n2 düğümlerinden olmak üzere 2 paket gelmektedir, burada Erlang hesaplamasını uygularsak Eşitlik 2'deki değer elde edilmektedir.

E=2 x 0,16=0,32E (Eş.2)

Call Blocking için bir ve iki paket geçişi durumları baz alınmış ve örnek sabit bir yol üzerinden işlem yapılmıştır. BSC-MAC protokolünde paketin izleyeceği yol belli olduğundan kök ve kaynak düğümler için geliştirilen algoritmalar çalıştırıldığında

diğer düğümler uyku modunda olacağından iletişim esnasında sadece bir yol açık olarak bulunacaktır. 100 byte'lık bir paket için h değerinin hesaplanması şu şekilde yapılmaktadır;

- Hattan 100 byte veri gelirse 100x8=800 bit veri transfer edilecek demektir.
- Ayrıca bant genişliği 10Kbyte=10240 bit değerine eşit olmaktadır.
- Dolayısıyla *h*=800/10240 =0,078 saniye çıkmaktadır.



Şekil 4.15. BSC-MAC protokolü örnek iletim

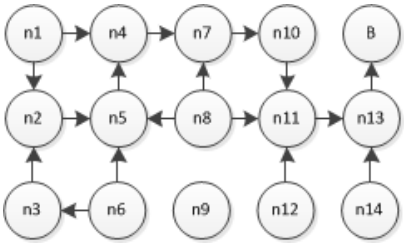
Şekil 4.15'de gösterildiği gibi bir paket için yani n3 düğümünden B düğümüne bir paket gönderilmesi durumunda 8 tane düğümün geçilmekte olduğu varsayılmıştır. Bundan dolayı her düğüm için işlem süresi 0,078 saniye olduğu için BSC-MAC protokolünde düğüm başına 1 paket için 1x0,078 = 0,078E değeri elde edilmektedir. Diğer düğümler BSC-MAC protokolünde kesin olarak uyku durumunda olacağı için ek bir paket yollama durumları olmayacaktır.

2 paket gönderimi için ise her düğümün bir pakette kendisinin yolladığı varsayılmıştır. Örneğin n5 düğümüne n2 düğümü hem n3 hem kendi ürettiği paketi yollayacaktır. Dolayısıyla n5 düğümünde 2x0,078=0,15E değeri elde edi1lecektir. n4 düğümü için ise n5 düğümünden gelen 2 paket ve kendi ürettiği paket olduğundan toplamda 3 paket gelecektir. Dolayısıyla 3x0,078=0,23E değeri elde edilecektir. BSC-MAC protokolünde 1 ve 2 paket için elde edilen değerler Çizelge 4.14'te gösterilmektedir.

Çizelge 4.14. BSC-MAC için düğüm bazından elde edilen erlang değeri

Düğüm Numarası / Gönderilen Paket Sayısı	n3	n2	n5	n4	n7	n10	n11
Bir paket(E)	0,078	0,078	0,078	0,078	0,078	0,078	0,078
İki paket(E)	0,078	0,15	0,23	0,31	0,39	0,46	0,54

Şekil 4.16'da S-MAC protokolü için BSC-MAC protokolü ile aynı senaryo üzerinden iletim şekli görünmektedir. S-MAC protokolünde belirlenen yol haricinde diğer yollarında uyanık olma ihtimali bulunmaktadır. En iyi ihtimal ile S-MAC protokolü BSC-MAC protokolü ile aynı değer çıkmaktadır. İletim anında periyodik uyumadan dolayı iletim hattına komşu olan düğümler uyku durumunda olursa BSC-MAC ile aynı çıkmaktadır. Burada maksimum durum değerlendirmesi yapılmaktadır. Bunun için periyodik uyuma ve uyanma esnasında iletim hattına komşu olan düğümlerin uyanık olduğu varsayılmıştır.



Şekil 4.16. S-MAC protokolü örnek iletim

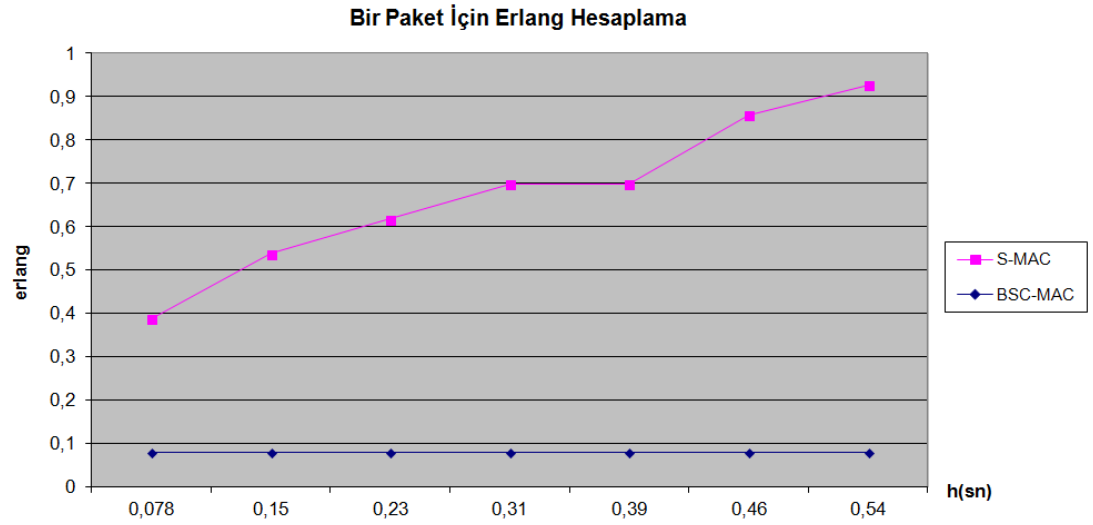
Şekil 4.16'da gösterildiği gibi bir paket için Şekil 4.15 ile aynı yolu izlediği varsayılarak paket gönderimi esnasında diğer düğümlerin uyanık olma durumları da hesaplamaya katılmıştır. Bunun için n2 düğümünden bir paket yollama durumunda 4 paket geçme ihtimali bulunmaktadır. Bunlar n3, n1 ve n6'dan gelebilmektedir. n6'dan gelecek olan paket n2 için 2 h süresi kadar süre geçirmektedir. Bu yüzden Erlang hesaplaması için n2'ye gelen paket sayısı 4 ve işlem süresi de 0,078 sn olarak bulunmuş olur. Dolayısıyla n2 için çıkan değer 4x0,078=0,31E olarak bulunmaktadır. n5 düğümü için n2'den 4 paket gelecektir, ayrıca n6 ve n8 düğümlerinden de birer paket geleceğinden 6 gelen paket olacak ve işlem süresi de

0,078 sn olacaktır. Dolayısıyla n5'te çıkan değer 6x0,078=0,46E olarak bulunmaktadır. Bu işlemler tüm düğümler için yapıldığında Çizelge 4.15'de iletim için belirlenen tüm yoldaki Erlang hesabı gösterilmektedir. İki paket gönderimi için senaryo yol üzerindeki düğümlerin ek bir paket yolladığı varsayılmıştır.

Çizelge 4.15. S-MAC için düğüm bazından elde edilen erlang değeri

Düğüm Numarası / Gönderilen Paket Sayısı	n3	n2	n5	n4	n7	n10	n11
Bir paket(E)	0,31	0,46	0,54	0,62	0,62	0,78	0,85
İki paket(E)	0,31	0,54	0,70	0,85	0,93	1,17	1,32

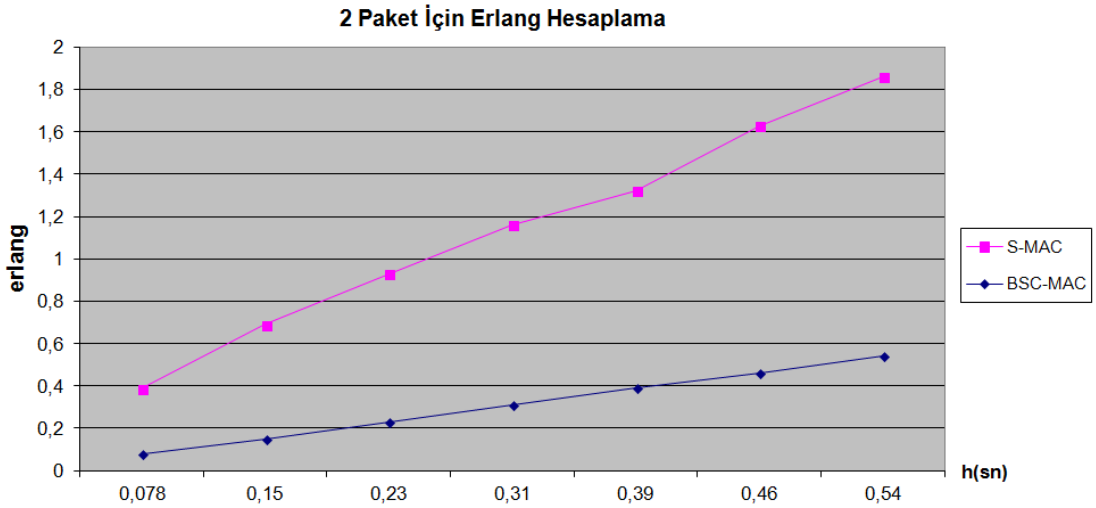
Şekil 4.17'de Çizelge 4.15'te elde edilen değerlerin grafik gösterimi yapılmaktadır. Burada S-MAC Düğümü minimum seviyede BSC-MAC düğümü ile aynı değerlere sahip olmaktadır. Ancak maksimum durumda Şekil 4.17'de gösterildiği gibi olmaktadır. Erlang Değeri ne kadar düşük olursa call blocking olma ihtimali o kadar azalmaktadır.



Şekil 4.17. Bir paket gönderimi için S-MAC ve BSC-MAC karşılaştırma

Şekil 4.18'de iki paket gönderimi esnasında elde edilen Erlang değerleri işlem süresi bazında gösterilmektedir. Burada iki paket anlamı yol üzerinde ki her paketin

gönderilen paket haricinde bir de kendi paketini oluşturup aynı yoldan göndermesi durumudur. Burada minumum durumda Erlang hesabı BSC-MAC protokolü ile aynı çıkmakta ancak çevre düğümlerinin açık olduğu düşünülecek olursa yani maksimum durumda Erlang hesaplaması Şekil 4.18'de gösterildiği gibi olmaktadır.

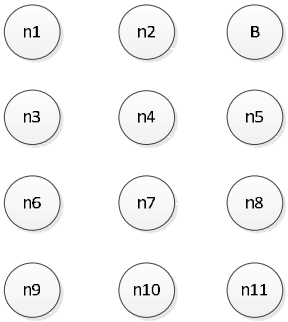


Şekil 4.18. İki paket için S-MAC ve BSC-MAC karşılaştırma

Elde edilen sonuçlara bakıldığında call blocking parametresi için BSC-MAC protokolünün S-MAC protokolünden daha iyi sonuç verdiği görülmektedir. S-MAC protokolünün en iyi durumu BSC-MAC protokolü ile aynı olmaktadır. Ancak S-MAC protokolünün iletişimi düşünüldüğünde aynı anda birçok komşu düğümün uyanık olma ihtimali çok yüksektir.

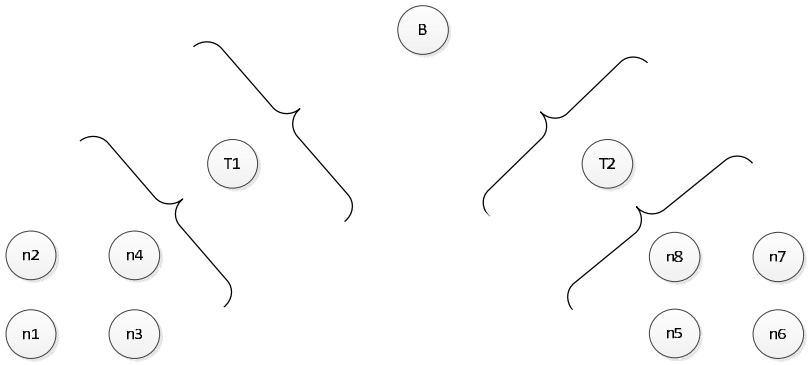
4.2. BSC-MAC Protokolünün Uygulaması

Uygulama TelosB düğümleri üzerinden gerçekleştirilmiştir. Elimizde bulunan 5 tane düğümün 2 tanesi kök düğüm olarak çalışmakta ve diğer 2 tanesi de kaynak düğüm olarak çalışmaktadır. Kök düğümlerin uyku durumuna geçmeleri işlemini tamamen baz istasyonu yönetmektedir. Kaynak düğümler ise birbirleri arasında adaletli iletişim mantığıyla çalıştıklarından birbirlerinin uyuma sürelerine kendileri karar vermektedir. Şekil 4.19'da örnek bir topoloji gösterilmektedir.



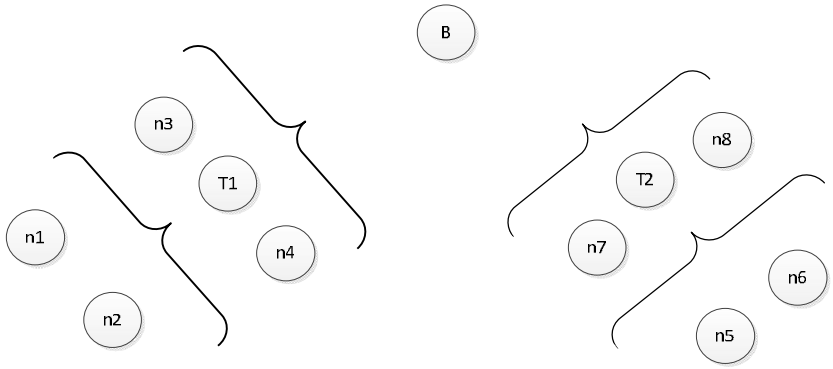
Şekil 4.19. Örnek topoloji

Şekil 4.19'de gösterilen topolojide B düğümü baz istasyonu olarak kabul edilmiştir. Dolayısıyla tüm düğümlerin amacı birbirleri üzerinden hangi yol olursa olsun baz istasyonuna mesaj yollamaktır. S-MAC protokolünde n9 düğümü baz istasyonuna mesaj yollamak isterse n10,n11,n8,n5 düğümlerini kullanabilir. Dolayısıyla periyodik uyumadan dolayı geçit düğümü olarak kullanılan bu düğümlerin uyku periyotları aynı zaman aralığına gelebilmektedir. BSC-MAC protokolünde ise kaynak düğümlerinin her zaman gönderebileceği bir yol kesinlikle bulunmaktadır. Şekil 4.20'de TinyOS iletim modeli gösterilmektedir.



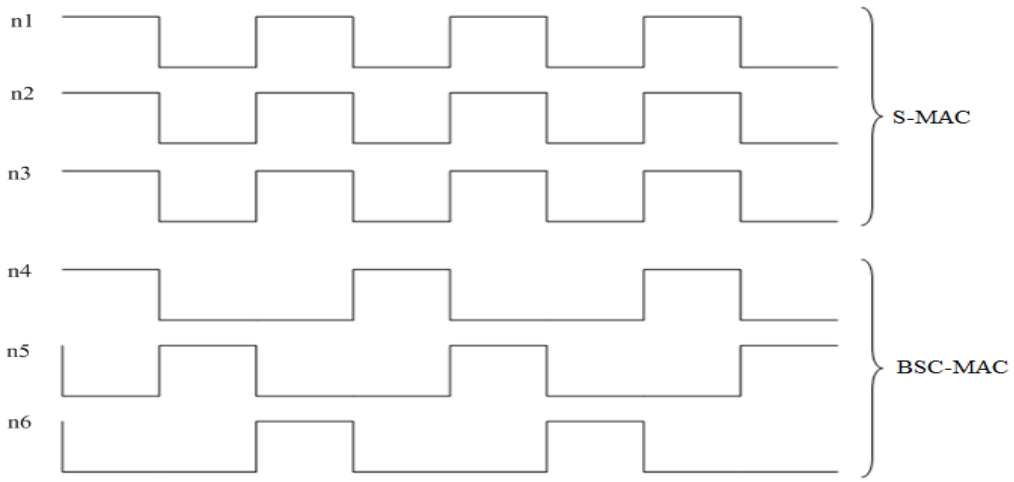
Şekil 4.20. TinyOS iletim modeli

Şekil 4.20'de n1,n2,n3,n4 düğümleri Baz istasyonu olan B düğümüne iletimi T1 düğümü üzerinde yapmaktadırlar. n5,n6,n7,n8 düğümleri ise baz istasyonu ile T2 düğümü üzerinden iletim yapmaktadırlar. Yani burada T1 ve T2 düğümleri tepebaşı olarak çalışmaktadırlar. Şekil 4.21'de ise BSC-MAC iletim modeli gösterilmektedir.



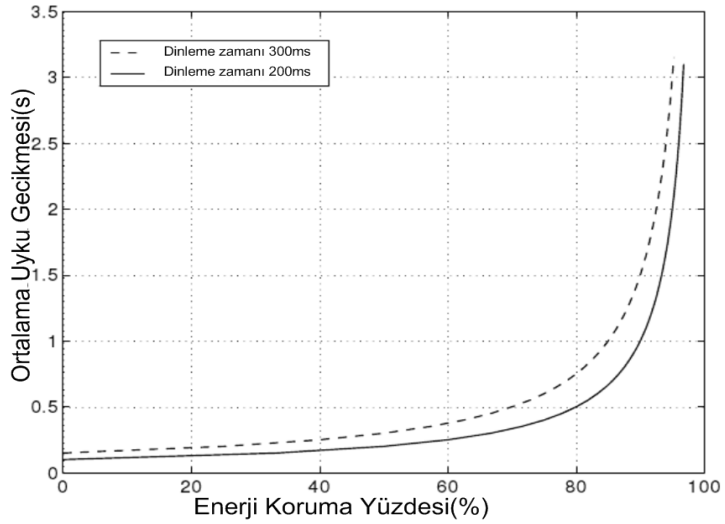
Şekil 4.21. BSC-MAC iletim modeli

Şekil 4.21'de tüm düğümlerin baz istasyonunu gördüğü varsayılmıştır. n1 ve n2 düğümleri baz istasyonu olan B düğümüne iletimi n3, T1, n4 düğümleri üzerinden yapabilmektedirler. n5 ve n6 düğümleri ise baz istasyonu ile n7, T2, n8 düğümleri üzerinden iletim yapabilmektedirler. Dolayısıyla tepebaşlarının enerjileri bitmiş olsa bile veya herhangi bir nedenden dolayı bu düğüm çalışmasa da diğer düğümler sayesinde iletişim devam edebilmektedir. Aynı zamanda bu tepebaşı olarak bulunan düğümler haricinde bu görevi gören diğer düğümler belirli koşullar örneğin kaynak düğümlerin enerjisinin bitmesi veya herhangi bir nedenden dolayı mesaj iletememesi durumunda kaynak düğüm olarak çalışmaktadırlar. Şekil 4.22'de S-MAC protokolünün periyodik uyuma uyanma stratejisi ve BSC-MAC protokolünün uyuma ve uyanma stratejisi gösterilmektedir [22].



Şekil 4.22. Yoğun trafik altında örnek gösterim

Şekil 4.22'de gösterildiği gibi S-MAC protokolünde tüm düğümler aynı anda uyumaktadırlar. Dolayısıyla düğümler uyuduğu zaman iletişim kesilmektedir. Buda hem gecikmeyi arttırmakta hem de iletişimin belirli zamanlarında durmasına neden olmaktadır. BSC-MAC protokolünde ise en az bir düğüm uyanık kaldığı için iletişim devamlı olmaktadır. Yoğun trafik anında iletişimin devamlı olması çok büyük önem arz etmektedir. BSC-MAC protokolünde bir düğüm uyanıp diğer düğüme mesaj göndermeden diğer düğüm uyku moduna geçmemektedir. Dolayısıyla hem düğümler arasında adaletli bir iletişim olmakta hem de gecikme olmamaktadır. Bunun yanında BSC-MAC protokolü enerji verimliliği konusunda S-MAC protokolünden daha iyi olduğu için ağın yaşam ömrü daha fazla artmakta ve düğümlerden yararlanma devamlı olmaktadır. Şekil 4.23'de S-MAC protokolü ile BSC-MAC protokolü arasındaki gecikme grafiğini göstermektedir.



Şekil 4.23. BSC-MAC ve S-MAC protokollerinin gecikme gösterimi

Şekil 4.23'de gösterildiği gibi BSC-MAC protokolü iletişim esnasında ve uyuyup uyanma esnasındaki gecikmede S-MAC düğümüne göre çok daha iyi sonuç vermektedir. Grafiğin bu şekilde çıkmasının sebebi is S-MAC protokolünün makalesinde gecikme için aldığı yöntemdir. Doğru bir karşılaştırma için S-MAC protokolündeki yöntem referans alınmıştır. S-MAC protokolünde gecikme düğümlerin periyodik uyuyup uyanma durumlarında ağın içerisinde kalan paketlerin

iletişim yapmamasından dolayı ortaya çıkan gecikme olarak değerlendirilmiştir. Örneğin A düğümü B düğümüne mesaj göndermeye çalışacak ancak B düğümü o anda uyuma modunda olacak dolayısıyla bu bir gecikmeye neden olmaktadır [22]. Şekil 4.23 Eşitlik 1, Eşitlik 2 ve Eşitlik 3 ile numaralandırılmış denklemler sayesinde elde edilmiştir.

$$D_s = T_{frame} / 2 \quad \text{(Eş.1)}$$

$$T_{frame} = T_{listen} + T_{sleep} \quad \text{(Eş.2)}$$

$$E_s = \frac{T_{sleep}}{T_{frame}} = 1 - \frac{T_{listen}}{T_{frame}} \quad \text{(Eş.3)}$$

BSC-MAC protokolünde ise ağda her zaman iletişim olduğundan gecikme ihtimali olmamaktadır. Bunun nedeni ise kök düğümlerinin birden fazla olmasından kaynaklanmaktadır. Kaynak düğümler mesajlarını muhakkak bir kaynak düğüm üzerinden baz istasyonuna göndermek durumundalar. S-MAC protokolündeki ara düğümlere benzemekte olan kök düğümlerinin işlevi ara düğümlerden daha farklıdır. Gerçekleştirilen uygulama sonuçlarının karşılaştırılması için S-MAC protokolü kullanılmıştır. S-MAC makalesinden alınan yaklaşık değerler ve işlemler aynen BSC-MAC içinde gerçekleştirilmiştir.

S-MAC MICA2 düğümler üzerinde çalışmaktadır. BSC-MAC ise TelosB düğümleri üzerinde çalışmaktadır. MICA2 düğümleri default olarak IEEE802.11 protokolünü kullanmakta iken TelosB düğümleri IEEE802.15.4 protokolünü kullanmaktadırlar [22]. S-MAC makalesi incelendiğinde karşılaştırmaların IEEE802.11 protokolü ile yapıldığı görülmüştür. TelosB düğümleri IEEE802.15.4 protokolünü kullandığından birebir karşılaştırma yapılamamaktadır.

IEEE 802.11 ve S-MAC protokollerinin uyku zamanında ve uyanık durumda iken yani paket gönderirken harcadıkları enerji sabittir. Buna göre karşılaştırma işlemleri aynı süre içerisinde aynı durumlarda işlemler yapılarak S-MAC protokolünün IEEE

802.11 protokolüne göre başarısı alınmıştır. Aynı işlemler BSC-MAC protokolü içinde kullanılmış ve alınan değerler Çizelge 4.16'da gösterilmektedir.

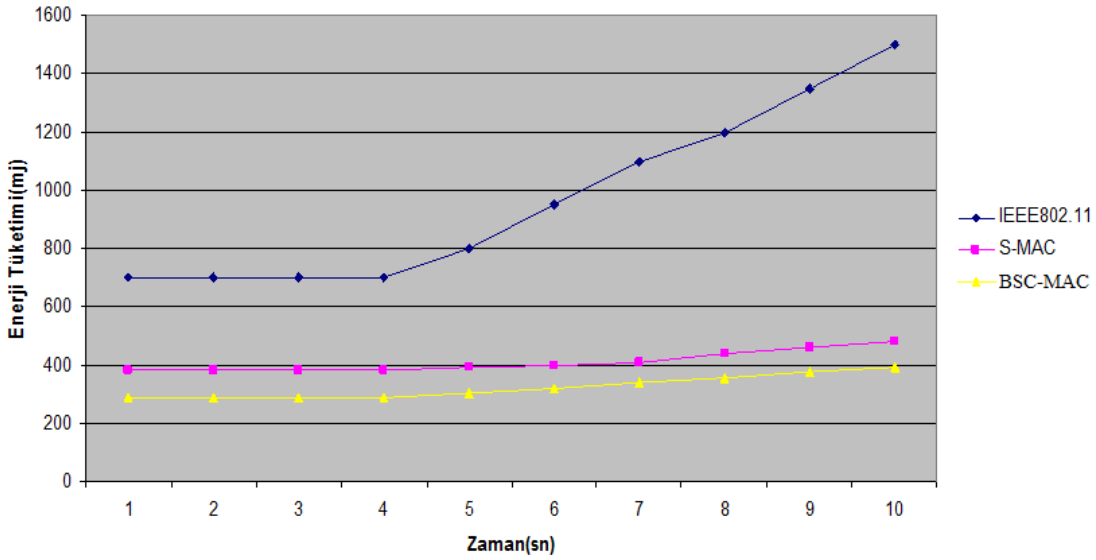
S-MAC protokolü yaptığı uygulama sonucunda elde ettiği sonuçlara bakıldığında toplamda 10 saniyelik bir iletişim yapmaktadır. Karşılaştırmanın doğru şekilde yapılması için S-MAC protokolündeki gibi n1 ve n2 kaynak düğümleri için uygulama BSC-MAC tarafından gerçekleştirilmiştir [22].

IEEE802.11 protokolü eğer ledler kapatılırsa iletim esnasında 19 mA uyku durumunda ise 0,1 µA harcamaktadır. Enerji verimli protokollerin ana amacı yoğun trafik altında enerji verimliliğini sağlayabilmektir. Karşılaştırmayı yapabilmemiz için S-MAC yönteminde uygulanan yöntem ve değerlerin aynısı kullanılmıştır. İşlemler için bir düğüm baz alınmıştır. Çünkü S-MAC protokolünde bu şekilde yapılmıştır. Zaman ayarlaması 30 ms olarak ayarlanmıştır. Trafiğin yoğun olduğu zamanlarda 15 µs uyku moduna geçmesi 15 µs uyanık kalması şeklinde ayarlanmıştır. Az trafikte uyanık 1 µs iken 29 µs uyku durumunda geçirmektedir. Çizelge 4.16'de gösterildiği gibi S-MAC protokolünün IEEE802.11 protokolüne göre saniye bazında kazançları gösterilmektedir.

Çizelge 4.16. Uygulama sonucu elde edilen sonuçların gösterimi

Protokol / Zaman(s)	**IEEE802.11(mj)**	**S−MAC(mj)**	**BSC-MAC(mj)**
1	700	380	285
2	700	380	285
3	700	380	285
4	700	380	285
5	800	390	303
6	950	400	321
7	1100	410	339
8	1200	440	357
9	1350	460	375
10	1500	480	393

Çizelge 4.16'da gösterilen 10 saniyelik iletişimde tüm düğümler çalıştığı için IEEE802.11'de tüketilen enerji S-MAC' ten daha fazla çıkmaktadır. S-MAC protokolü ise her saniye içerisinde düğümleri mutlaka çok az da olsa uyku durumuna geçirdiği için tüketilen enerji miktarı IEEE802.11'den daha iyi çıkmaktadır. Yani yoğun trafikte S-MAC IEEE802.11'den ortalama %46 daha verimli çalışırken diğer zamanlarda trafik yoğunluğuna göre farklı uyuma süresi uyguladığından trafiğin az olduğu durumlarda enerji verimliliği açısından %68'e kadar tasarruf edebilmektedir. S-MAC için yapılan işlemler BSC-MAC içinde yapıldığında IEEE802.11 protokolüne göre elde edilen kazancın S-MAC'ten daha iyi çıktığı görülmektedir. Şekil 4.25'de uygulama sonuçları grafiksel olarak gösterilmektedir.

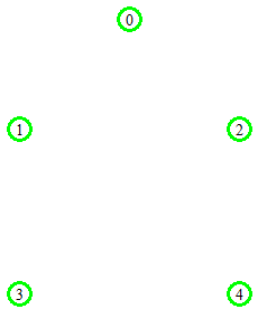


Şekil 4.24. Uygulama sonuçlarının karşılaştırılması

Şekil 4.24'da görüldüğü gibi BSC-MAC protokolünün S-MAC protokolünden her koşulda daha iyi sonuç verdiği görülmektedir. Tabi ki burada sadece bir düğüm üzerinden yapılan karşılaştırmada çok fazla bir verimlilik olmadığı düşünülse bile geniş ölçekli ve çok fazla düğümün olduğu durumlarda enerji verimliliğinin S-MAC protokolünden ve diğer karşılaştırılan protokollerden daha iyi çıkacağı ispatlanmaktadır.

4.3. Uygulama İle Benzetimin Doğrulaması

BSC-MAC için hem benzetim hemde uygulama gerçekleştirilmiştir. Dolayısıyla uygulama sonuçlarıyla benzetim sonuçlarının doğrulaması yapılarak BSC-MAC protokolünün davranışları bu bölümde incelenmiştir. Şekil 4.25'te doğrulama için kullanılan topolojinin ns-2'dan alınmış grafiği gösterilmektedir.



Şekil 4.25. Doğrulama için kullanılan topoloji

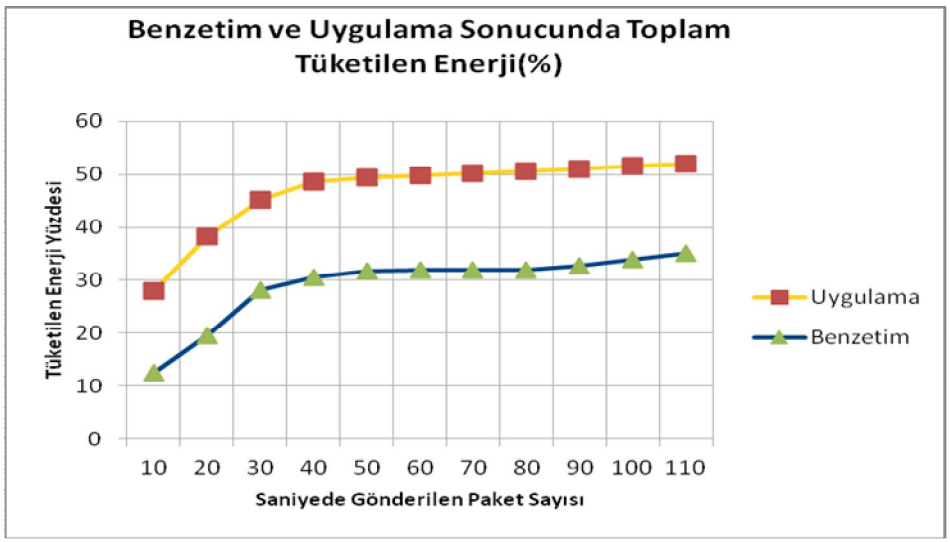
Şekil 4.25'te gösterilen grafikte 3 ve 4 numaralı düğümler baz istasyonu olan 0 numaralı düğümü kapsamamaktadırlar. 1 ve 2 numaralı düğümler ise baz stasyonu olan 0 numaralı düğümü kapsamaktadırlar. Kök ve kaynak ayırma algoritması çalıştırıldığında 1 ve 2 numaralı düğüm kök düğüm olmakta iken 4 ile 5 numaralı düğümlerde kaynak düğüm olmaktadır.

Uygulamadaki gereksinimlerden dolayı topoloji ufak tutulmuştur. Benzetim ve uygulama 100 saniye çalıştırılmıştır. Başlangıç enerjileri 100 joule olarak ayarlanmıştır. Farklılığın oluşabilmesi için düğümler saniyede 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 ve 110 paket üretip gönderme işlemi gerçekleştirmiştir. Çizelge 4.17'de gerçekleştirilen benzetim ve uygulama sonucunda elde edilen düğümlerde tüketilen enerji değerleri gösterilmektedir

Çizelge 4.17. Doğrulama için tüketilen enerji miktarı

Paket Sayısı / Parametre	10	20	30	40	50	60	70	80	90	100	110
Benzetim(%)	12,59	19,48	28,15	30,5	31,68	31,79	31,8	31,84	32,69	33,91	35,07
Uygulama(%)	27,8	38,2	45,3	48,7	49,5	49,9	50,3	50,53	51,1	51,6	52,05

Hem uygulama hemde benzetim için BSC-MAC 100 saniye çalıştırılmış ve saniyede 10 paket üretiminden başlayıp saniyede 110 paket üretimine kadar olan her üretim şekli için elde edilen sonuçlar Çizelge 4.17'de gösterilmiştir. Şekil 4.26'da ise Çizelge 4.17'de elde edilen sonuçların grafiksel gösterimi yapılmaktadır.



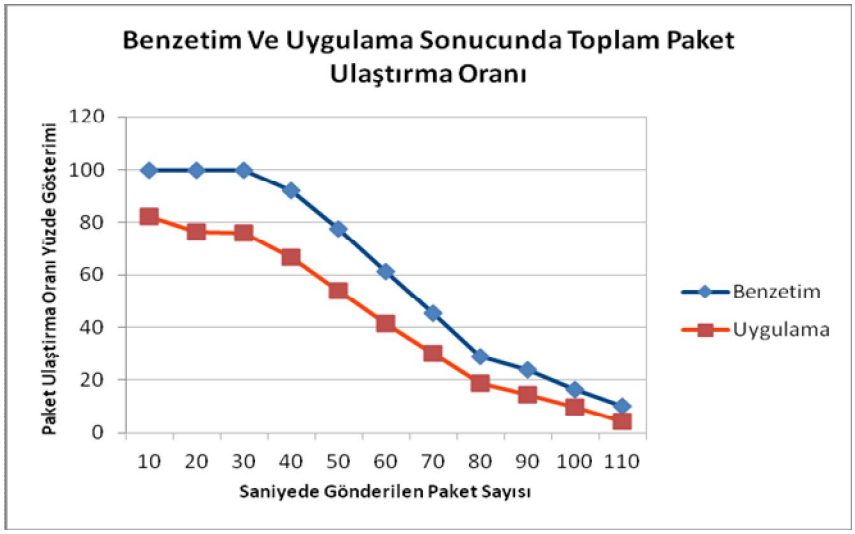
Şekil 4.26. Doğrulama için tüketilen enerji miktarı gösterimi

Şekil 4.26'dan anlaşılacağı gibi uygulama ve benzetim sonuçları parelelik arz etmektedir. Bu değerlerin farklı çıkması ise uygulama ile benzetim ortamlarında kullanılan bazı harici parametrelerin birbirlerinden farklı olmasından kaynaklanmaktadır. Çizelge 4.18'de ise doğrulama için paket ulaştırma işleminde elde edilen sonuçlar gösterilmektedir.

Çizelge 4.18. Doğrulama için paket ulaştırma oranı

Paket Sayısı / Parametre	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**	**110**
Benzetim(%)	99,7	99,78	99,91	92,04	77,55	61,21	45,08	29,02	23,7	16,4	10,2
Uygulama(%)	82,4	76,6	76	66,72	53,7	41,52	30,02	18,8	14,52	9,93	4,47

Çizelge 4.18'de kaynak ve kök düğümler tarafından üretilip hedef düğüm olan baz istasyonuna gönderilen paketlerin hedefe ulaştırılma oranları yüzdesel olarak gösterilmektedir. Saniyede 10 paket üreten her düğümün kuyruğu saniyede 110 paket üreten her düğümden daha fazla paketi baz istasyonuna ulaştırmaktadır. Buradaki kayıpların nedenleri ise düğümümlerin gönderdikleri paketlerin çakışmasından, kuyruğun dolmasından dolayı tıkanıklığın meydana gelmesi gibi durumlardan dolayı oluşmaktadır. Şekil 4.27'de Çizelge 4.18'de elde edilen sonuçların grafiksel gösterimi yapılmaktadır.



Şekil 4.27. Doğrulama için paket ulaştırma oranı gösterimi

Şekil 4.27'de gösterilen grafikte uygulama ve benzetim için gerçekleştirilen işlemlerde elde edilen sonuçların tam olmasada birbirleriyle parelellik arz ettiği görülmektedir. Saniyede 10 paket gönderirken elde edilen hedefe ulaştırma oranı yüksek iken saniyede 110 paket gönderimi sırasında hedefe paket ulaştırma oranı paket çakışmaları ve kuyruğa alınmama gibi nedenlerden dolayı azaldığı

görülmektedir. Hem paket ulaştırma oranına hem kalan enerji durumlarına bakıldığında elde edilen parelellik ten dolayı uygulama ve benzetimde çalıştırılan BSC-MAC protokolünün başarılı olduğu görülmektedir.

5. SONUÇLAR ve ÖNERİLER

Kablosuz algılayıcı ağların topolojilerinin oluşturulması genelde rastgele olmaktadır. Milyonlarca düğümün kolay ulaşılamayacak yerlere atılması sonucu ortaya çıkan en önemli problem bu düğümlerin pillerinin şarj edilememesi ve değiştirilememesidir. Düğümlerin ömrünün tükenmesi ağın ömrünü direk etkilemektedir. Gecikme gibi kablosuz ağlarda önemli olan konular algılayıcı ağlarda geri plana düşmektedir. Algılayıcı ağlardaki düğümlerin bazılarının veya kritik noktada bulunan bazı düğümlerin pillerinin bitmesi iletişimin kesilmesine neden olabileceği gibi düğümlerin ömrünün ve doğru orantılı olarak ağın ömrününde çok hızlı bir şekilde bitmesi durumunu ortaya çıkaracaktır. Bu yüzden algılayıcı ağlarda enerji verimliliği çok önemli bir unsurdur. Bu yüzden bu konunun geliştirilmesi ve optimum seviyeye çıkarılması gerekmektedir. Algılayıcı ağlarda MAC katmanında ortaya çıkan enerji israfı genelde ortamın gereksiz yere dinlenmesi, kontrol paketlerinin fazlalığı, istem dışı alım, paket çakışması olarak sınfılandırılabilir.

BSC-MAC protokolünün amacı ağdaki düğümlerin enerjilerini optimum seviyede kullanabilmelerini sağlamak ve böylece ağın ömrünü ve düğümlerden yararlanma süresini uzatmaktır. Bunun için BSC-MAC düğümlerin uyarlamalı olarak uyumasını sağlamakta ve düğümlerin enerjilerini en iyi şekilde kullanabilmelerine izin vermektedir. BSC-MAC protokolünde enerji tasarrufu, boşta kalan düğümlerin uyku durumuna geçirilmesi ve her düğüme adaletli erişim verilerek sağlanmıştır. BSC-MAC uyku durumuna geçiş işlemini uyarlamalı olarak gerçekleştirmektedir. BSC-MAC protokolünde enerji tasarrufu ve adaletli erişim için iki farklı mekanizmayla iletişim gerçekleştirilmektedir. Bunlar kök ve kaynak düğüm olarak adlandırılmıştır. Kaynak düğümler verileri kök düğümler üzerinden baz istasyonuna iletirler. BSC-MAC protokolünde kök düğümler ve kaynak düğümler kendileriyle ve birbirleri ile olan iletişimi gerçekleştirebilmek için uyuma periyotlarının senkronizasyonunu tam olarak yapabilmelidirler. Bunun için uyarlamalı olarak iki farklı yöntem kullanılmaktadır. Kök ve kaynak düğümlerin belirlenmesi işlemi baz istasyonu tarafından gerçekleştirilmektedir. Baz istasyonu kök düğümlerin uyuma zamanlarını

ve onların bilgilendirme işlemi ile senkronizasyonunu direk gerçekleştirmektedir. Kaynak düğümlerde ise baz istasyonu tarafından kaynak düğüm olarak atandıktan sonra kendi aralarında yolu ilk ele geçiren düğüm diğer düğümleri uyku kipine geçirmektedir. Kök ve kaynak düğümlerin uyku planları ile senkronizasyonu en başta baz istasyonu tarafından gerçekleştirildiği için optimum eşleme gerçekleştirilirmektedir. Bu sayede hem boşta olan düğümlerin uyku kipine gönderilmesi hemde düğümlerin ortama adaletli erişimi sayesinde var olan protokollerden daha iyi bir iletişim ve enerji verimliliği sağlanmaktadır.

BSC-MAC protokolünün hem benzetim hemde uygulaması gerçekleştirilmiştir. Benzetimde BS-MAC protokolü S-MAC, P-MAC, AEEMAC protokolleriyle karşılaştırılmıştır. Çizelge 5.1'de karşılaştırma parametreleri bazında BSC-MAC protokolünün diğer protokollere göre ortaya çıkan kazancı yüzdesel olarak gösterilmiştir. Çizelge 5.1'de gösterildiği gibi BSC-MAC protokolünün diğer protokollerden daha başarılı olduğu görülmektedir. Izgara topolojide toplam kalan enerji parametresinde AEEMAC protokolünden daha düşük bir kazanç elde edilmiştir. Bunun nedeni ise BSC-MAC protokolünün paket ulaştırma oranının AEEMAC protokolünden daha yüksek çıkmasından dolayı daha fazla enerji harcamasıdır.

Çizelge 5.1. BSC-MAC protokolünün benzetimde elde edilen kazancı

Parametreler	**Topoloji**	**S-MAC**	**AEEMAC**
Toplam Kalan Enerji(%)	Doğrusal	11,72	10,67
	Izgara	34,35	-0,28
Uctan Uca Gecikme(%)	Doğrusal	17,53	17,81
	Izgara	38,48	8,65
Paket Ulaştırma Oranı(%)	Doğrusal	0,25	1,78
	Izgara	269,94	56,73
Üretilen İş(%)	Doğrusal	1,05	2,13
	Izgara	366,66	66,97
Bant Genişliğinden Yararlanma(%)	Farklı	1,46	Yok

Uygulamada ise BS-MAC protokolü S-MAC ve IEEE 802.11 protokolleriyle karşılaştırılmıştır. Çizelge 5.2'de BSC-MAC protokolünün tüketilen enerji

parametresi bazında S-MAC ve IEEE 802.11 protokolleriyle karşılaştırma işleminde elde edilen kazancı gösterilmektedir. Çizelge 5.2'de gösterildiği gibi BSC-MAC protokolünün diğer protokollerden daha başarılı olduğu ve düğümlere enerji tüketiminde yüzdesel olarak daha fazla kazanç sağladığı görülmektedir.

Çizelge 5.2. BSC-MAC protokolünün uygulamada elde edilen kazancı

Zaman(sn)	IEEE 802.11(%)	S-MAC(%)
1	59	24
2	59	24
3	59	24
4	59	24
5	62	22,25
6	66	19
7	69	17
8	70	18
9	72	18
10	73	18

BSC-MAC protokolü var olan protokollerden daha iyi sonuç verdiği elde edilen sonuçlardan görülmektedir. Ancak BSC-MAC protokolünün baz istasyonu tarafından gerçekleştirilen ve diğer düğümlerin senkronizasyonlarının sağlanması için kullanılan süre atamasında gerçekleştirilecek bir optimizasyonun bu protokolü daha fazla geliştireceği düşünülmektedir. Ayrıca ölçeklenebilirliği arttıracak çalışmaların bu protokolü daha fazla geliştireceği ve daha iyi sonuçların elde edileceği öngörülmektedir.

BSC-MAC protokolü S-MAC, P-MAC, AEEMAC protokolleri ile birçok parametre üzerinden karşılaştırılmıştır ve BSC-MAC protokolünün diğer protokollerden enerji korunumu, ortama adaletli erişim, iletişimin devamlılığı ve tüm ağın ömrünün uzaması durumlarında daha iyi sonuç verdiği elde edilen sonuçlardan anlaşılmıştır.

KAYNAKLAR

1. Shen, C., Sriathapornphat, C. ve Jaikaeo, C., "Sensor Information Networking Architecture and Applications", **IEEE Pers. Commn,** 52 – 59, 2001.

2. Blumenthal, J. ve Handy, M., "Wireless Sensor Networks - New Challenges in Software Engineering", **9th IEEE International Conference on Emerging Technologies and Factory Automation,** Vol. 1, 551-556, 2003.

3. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. ve Cayirci, E., "Wireless Sensor Networks: A Survey", **Computer Networks,** Vol. 38, 393-422, 2002.

4. Pottie, G. J. ve Kaiser, W. J., "Wireless Integrated Network Sensors", **Commun. ACM,** Vol. 43, 551 – 558, 2000.

5. Kleinrock, L. ve Tobagi, F.A., "Packet switching in radio channels: Part I—Carrier sense multiple-access modes and their throughput-delay characteristics", **IEEE Transactions on Communications,** Vol.23(12), 1400–1416, 1975.

6. Lauwens, B., Scheers, B. ve Van de Capelle, A., "Performance analysis of unslotted CSMA/CA in wireless networks", **Telecommunication Systems,** Vol. 44(1-2), 109-123, 2010.

7. Gao, B., He, C. ve Jiang, L.G., "Modeling and Analysis of IEEE 802.15.4 CSMA/CA with Sleep Mode Enabled", **2008 11th IEEE Singapore International Conference On Communications Systems (ICCS),** VOL. 1-3, 6-11, 2008.

8. Kwon Y., Chae Y.,"Traffic Adaptive IEEE 802.15.4 MAC for Wireless Sensor Networks", **Embedded and Ubiquitous Computing Lecture Notes in Computer Science,** Vol. 4096, 864-873, 2006.

9. Huang, Y.K., Huang, S.W. ve Pang, A.C., "An Energy-Efficient MAC Design for IEEE 802.15.4-Based Wireless Sensor Networks", **IFIP International Conference Embedded and Ubiquitous Computing**, Vol. 4809, 237-248, 2007.

10. Xu, Q., Rong, L. ve Fang, S.A., "Energy-efficient Scheme for IEEE 802.15.4 Compliant Device", **Progress in Electromagnetics Research Symposium**, 353-356, 2009.

11. Cho, D.H., Song, J.H. ve Han, K.J., "An Adaptive Energy Saving Mechanism for the IEEE 802.15.4 LR-WPAN", **1st International Conference on Wireless Algorithms, Systems and Applications**, Vol. 4138, 38-46, 2006.

12. Alavi, S. M. M., Walsh, M. J. ve Hayes, M. J., "Robust distributed active power control technique for IEEE802.15.4 wireless sensor networks — A quantitative feedback theory approach", **Control Engineering Practice,** Vol. 17(7), 2009.

13. Misic, J., "Algorithm for equalization of cluster lifetimes in a multi-level Beacon enabled 802.15.4 sensor network", **Computer Networks,** Vol. 51(11), 3252-3264, 2007.

14. Kredo, K. ve Mohapatra, P., "Medium access control in wireless sensor networks", **Computer Networks,** Vol. 51(4), 961–994, 2007.

15. Singh, S. ve Raghavendra, C., "PAMAS – power aware multiaccess protocol with signaling for ad hoc networks", **SIGCOMM Computer Communications Review,** Vol. 28 (3), 5–26, 1998.

16. Vuran, M.C. ve Akyildiz, I.F., "Spatial correlation-based collaborative medium access control in wireless sensor networks", **IEEE/ACM Transactions on Networking**, Vol.14 (2), 316–329, 2006.

17. Schurgers, C., Tsiatsis, V., Ganeriwal, S. ve Srivastava, M., "Optimizing sensor networks in the energy-latency-density design space", **IEEE Transactions on Mobile Computing,** Vol.1(1), 70–80, 2002.

18. Lin, E.-Y.A., Rabaey, J.M. ve Wolisz, A., "Power-efficient rendezvous schemes for dense wireless sensor networks", **Proceedings of the IEEE International Conference on Communications (ICC)**, Vol. 7, 3769–3776, 2004.

19. Polastre, J., Hill, J. ve Culler, D., "Versatile low power media access for wireless sensor networks", **Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys),** 95–107, 2004.

20. El-Hoiydi, A. ve Decotignie, J.-D., "WiseMAC: an ultra low power MAC protocol for multi-hop wireless sensor networks", **Proceedings of the International Workshop on Algorithmic Aspects of Wireless Sensor Networks (Algosensors),** 18–31, 2004.

21. S Mahlknecht, S. ve Böck, M., "CMSA-MPS: a minimum preamble sampling MAC protocol for low power wireless sensor networks", **Proceedings of the IEEE International Workshop on Factory Communication Systems**, 73–80, 2004.

22. Ye, W., Heidemann, J. ve Estrin, D., "An energy-efficient MAC protocol for wireless sensor networks", **Proceedings of the Joint Conference of the IEEE Computer and Communications Societies (InfoCom)**, Vol. 3, 214–226, 2002.

23. Van Dam, T. ve Langendoen, K., “An adaptive energy-efficient MAC protocol for wireless sensor networks”, **Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys)**, 171–180, 2003.

24. Zheng, T., Radhakrishnan, S. ve Sarangan, V., "PMAC: An adaptive energy-efficient MAC protocol for wireless sensor networks", **Proceedings of the IEEE International Parallel and Distributed Processing Symposium,** 65–72, 2005.

25. Lin, P., Qiao, C. ve Wang, X., “Medium access control with a dynamic duty cycle for sensor networks”, **Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC),** Vol. 3, 1534–1539, 2004.

26. Rajendran, V., Obraczka, K. ve Garcia-Luna-Aceves J., “Energy efficient, collision-free medium access control for wireless sensor networks”, **Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys),** 181–192, 2003.

27. Abramson, N., “The ALOHA System – Another Alternative for Computer Communications”, **Proceedings Fall Joint Computer Conference**, Vol. 37, pp. 281-285, 1970.

28. Buettner, M., Yee, G., Anderson, E. ve Han, R., “X-MAC: A Short Preamble mac Protocol for Duty-Cycled Wireless Sensor Networks”, **Proc. First ACM Conf. Embedded Networked Sensor Systems(SenSys),** 2006.

29. Rhee, I., Warrier, A., Aia, M. ve Min, J., “Z-MAC: a hybrid MAC for wireless sensor networks”, **Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys),** 90–101, 2005.

30. Sha, L., Kai-Wei, F. ve Prasun, S., " CMAC: An Energy Efficient MAC Layer Protocol Using Convergent Packet Forwarding for Wireless Sensor Networks", **Journal ACM Transactions on Sensor Networks (TOSN),** Vol.5(4), 2009.

31. Nait-Abdesselam, F., Bensaou, B., Soete, T. ve Hung, K-L., "O-MAC: An Organized Energy-Aware MAC Protocol for Wireless Sensor Networks", **Communications, 2007. ICC '07. IEEE International Conference on,** 3648 - 3653 , 2007.

32. Rajendran, V., Garcia-Luna-Aveces, J.J. ve Obraczka, K., "FLAMA:Energy-efficient, application-aware medium access for sensor networks", **Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on**, 625 - 630, 2005.

33. Sitanayah, L., Sreenan, C.J. ve Brown, K.N., "ER-MAC: A Hybrid MAC Protocol for Emergency Response Wireless Sensor Networks", **Sensor**

Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on, 244 - 249, 2010.

34. Roy, A. ve Sarma, N.,"AEEMAC: Adaptive energy efficient MAC protocol for wireless sensor networks", **India Conference (INDICON), 2011 Annual IEEE,** 1-6, 2011.

35. Chatzigiannakis, I., Kinalis, A. ve Nikoletseas, S., "Wireless sensor networks protocols for efficient collision avoidance in multi-path data propagation", **Proceedings of the ACM International Workshop on Performance Evaluation of Wireless Ad Hoc**, 8–16, 2004.

36. Chatzigiannakis, I., Dimitriou, T., Mavronicolas, M., Nikoletseas, S. ve Spirakis, P., "A comparative study of protocols for efficient data propagation in smart dust networks", **Parallel Processing Letters,** Vol.13 (4), 615–627, 2003.

37. Karagiannis, A., Kokkorikos, S. ve Constantinou, P., "Energy Consumption Analysis and Optimization techniques for Wireless Sensor Networks", **ISCC: 2009 IEEE Symposium On Computers and Communications,** Vol.1(2), 8-14, 2009.

38. Zebbane, B., Chenait, M., Badache, N. ve Zeghilet, H., "Topology Control Protocol for Conserving Energy in Wireless Sensor Networks", **ISCC: 2009 IEEE Symposium On Computers And Communications,** Vol.1(2), 716-719, 2009.

39. Jabbar, S., Minhas, A.A., Akhtar, R.A. ve Aziz, M.Z., "REAR: Real-time Energy Aware Routing for Wireless Adhoc Micro Sensors Network", **Eighth IEEE International Conference On Dependable, Autonomic And Secure Computing, Proceedings,** 825-830, 2009.

40. Internet: Crossbow Technology, Inc. Home Page, http://www.xbow.com, 2010.

41. Internet: TinyOS, Home Page, http://www.tinyos.net/special/mission, 2010.

42. Internet: Stanford University, Home Page, http://sing.stanford.edu/doc/tinyos-alliance.pdf, 2011.

43. Internet: Stanford University, Home Page, http://csl.stanford.edu/~pal/pubs/ tinyos-programming.pdf, 2011.

44. Internet: Berkeley University, Home Page, http://www.cs.berkeley.edu/~pal /pubs/nido.pdf, 2011.

45. Internet: Berkeley University, Home Page, http://www.cs.berkeley.edu/~pal/ research/ tossim. html, 2011.

46. Landsiedel, O., Wehrle, K. ve Götz, S., "Accurate Prediction of Power Consumption in Sensor Networks", **Embedded Networked Sensors,** 2005. EmNetS-II. The Second IEEE Workshop on, 37-44, 2005.

47. Shnayder, V., Hempstead, M., Chen, B., Allen, G.W. ve Welsh, M., "Simulating the power consumption of large-scale sensor network applications", **ACM Press,** 188—200, 2004.

48. Gay, D., Levis, P., von Behren, R., Welsh, M., Brewer, E. ve Culler, D., "The nesC language: A holistic approach to networked embedded systems", **In Proc. Programming Language Design and Implementation (PLDI),** 2003.

49. Athanassoulis, M., Alagiannis, I. ve Hadjiefthymiades, S., "Energy efficiency in wireless sensor networks: A utility-based architecture", **13th European Wireless 2007**, 2007.

50. Zhao, F. ve Guibas, L., "Wireless Sensor Networks: An Information Processing Approach", **Elsevier,** Morgan Kaufmann Publishers, 2004.

51. Girod, L., Elson, J., Cerpa, A., Stathopoulos, T., Ramanathan, N. ve Estrin, D., "EmStar: A software environment for developing and deploying wireless sensor Networks", **In Proc. USENIX'04,** 2004.

52. Sundresh, S., Kim, W. ve Agha. G., "SENS: A Sensor Environment and Network Simulator", **In Proc. of Annual Simulation Symposium",** 2004.

53. Internet: TinyOS, Document, http://www.tinyos.net/tinyos-2.x/doc/, 2011.

54. Erlang, A.K., "Solutions of some problems in the theory of probabilities of significance in automatic telephone exchanges", **The Post Office Elecrr. Eng. J.,** Vol. 10, pp. 189-197, 1918.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: TOKLU, Sinan
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 01.05.1979 Adana/Ceyhan
Medeni hali	: Bekâr
Telefon	: 0 (312) 203 8073
e-mail	: sinan.toklu@teias.gov.tr, sinantoklu@gmail.com

Eğitim

Derece	**Eğitim Birimi**	**Mezuniyet tarihi**
Yüksek lisans	Gazi Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği	2007
Lisans	Doğu Akdeniz Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği	2004

İş Deneyimi

Yıl	**Yer**	**Görev**
2004 – 2005	COMTECH Telekominikasyon	Network Mühendisi
2005 – 2006	Başbakanlık Özürlüler İdaresi	Çözümleyici
2006 – 2010	Gazi Üniversitesi	Araştırma Görevlisi
2010 – (Devam)	TEİAŞ	Mühendis

Sertifikalar

1. Comtech Uzak Mesafe Telefon Hizmetleri Eğitim Sertifikası.
2. MCSE Eğitimini Tamamlama Sertifikası.
3. MCP(Microsoft Certified Professional) Sertifikası 70-215 (Installing, Configuring

and Administering Microsoft Windows 2000 Server).

4. Probil e-öğrenme içerik geliştirme ve proje yönetimi sertifikası.
5. CISCO CCNA Eğitim Katılım Sertifikası.
6. JAVA ve Nesneye Dayalı Programlama Eğitim Katılım Sertifikası.
7. JAVA EE Eğitim Katılım Sertifikası.
8. Linux Eğitim Sertifikası.

Yabancı Dil

İngilizce

Yayınlar

1. **Toklu, S.,** Akcayol, M.A., "Congestion Control In WAP Traffic and Transport Layer Protocols",**J. Fac. Eng. Arch. Gazi Univ.**, Vol.24(3), pp.397-408, 2009.
2. Dener, M., Akcayol, M.A., **Toklu, S.,** Bay, Ö.F., "Genetic Algorithm Based a New Algorithm for Time Dynamic Shortest Path Problem", **J. Fac. Eng. Arch. Gazi Univ.**,Vol.26(4), 2011.
3. Dener, M., **Toklu, S,** "Performance Evaluation of DSDV and DSR Manet Routing Protocols", **Gazi University Journal of Polytechnic**, Vol. 12 (3), pp.157-166, 2009.
4. **Toklu, S.,** Erdem, A.O., "Performance Evaluation of IDLE and POWER_DOWN Power Mode", **International Journal of Informatics Technologies,** Vol. 5(3), pp.29-34, 2012.
5. Akcayol, M.A., Şimşek, M., Bay, İ., **Toklu, S.,** Doğru, I.A., "Genetic Algorithm Based Location Optimization of Emergency Service Units", 4th FAE International Symposium, European University of Lefke, 2006.
6. **Toklu, S.,** Simsek, M., Yildiz, O., Bay, İ., Simsek, A., Akcayol, M.A., "A Priority Based Protocol and Load Balancing for Queue Management on Wireless Networks", The 2008 International Conference on Wireless Networks (ICWN'08), pp.341-344, July 14-17, 2008, Monte Carlo Resort, Las Vegas, Nevada, USA.
7. Dogru, I. A., Simsek, M., **Toklu, S.,** Yildiz, O., Akcayol, M.A., "Rule-Based Mobility Management Routing for Mobile Ad Hoc Networks", The 2008

International Conference on Wireless Networks (ICWN'08), pp.175-179, July 14-17, 2008, Monte Carlo Resort, Las Vegas, Nevada, USA.

8. Simsek, M., **Toklu, S.,** Akcayol, M.A., Soncul, H., Ergun, M.A., "A Comprehensive Analysis of E-Government Studies in Turkey, Problems and Suggestions", The IADIS WWW/Internet 2009 Conference, November 19-22, 2009, Rome, Italy.