YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TinyOS İŞLETİM SİSTEMİNİN VF-1A GENEL AMAÇLI TELSİZ ALGILAYICI DÜĞÜM ÜZERİNE UYARLANMASI

Bilgisayar Mühendisi Haluk ÖRNEK

FBE Bilgisayar Mühendisliği Anabilim Dalında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Yrd. Doç. Dr. A. Gökhan YAVUZ

İSTANBUL, 2009

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ vi

ÇİZELGE LİSTESİ viii

ÖNSÖZ ix

ÖZET x

ABSTRACT xi

1. GİRİŞ 1

2. TELSİZ ALGILAYICI AĞLARIN GELİŞİMİ 3

3. KULLANIMDA OLAN TELSİZ ALGILAYICI AĞLAR 10

3.1 Great Duck Island 10
3.2 Hogthrob 12
3.3 ZebraNET 14
3.4 Code Blue 16

4. KULLANIMDA OLAN TELSİZ ALGILAYICI DÜĞÜMLER 20

4.1 Wec, Rene, Rene2, Dot 20
4.2 Mica, Mica2Dot, Mica2, Telos 22
4.3 XYZ 24
4.4 Mulle 25
4.5 UIUC – Sensor Node 26
4.6 Eyes 26
4.7 Tmote Invent 27
4.8 MicaZ 27

5. DÜĞÜMLER İÇİN GELİŞTİRİLMİŞ İŞLETİM SİSTEMLERİ 28

5.1 TinyOS 28
5.2 MANTIS 32
5.3 SOS 35
5.4 Ticari İşletim Sistemleri 36
5.5 Olaya Dayalı ve Çoklu Görevlendirmeli İşletim Sistemleri Karşılaştırması 37

6. İŞLETİM SİSTEMİNİN ÇALIŞACAĞI DÜĞÜMÜN DONANIM ELEMANLARI 41

6.1 Mikrodenetleyici Birimi 41
6.2 Telsiz İletişim Birimi 41
6.3 İkincil Bellek Birimi 42
6.4 Güç Yönetim Birimi 42

6.5 Algılayıcılar ... 42
7. İŞLETİM SİSTEMİNİN SEÇİMİ VE UYARLANMASI ... 49
7.1 Telsiz Algılayıcı Ağ Üzerinde Çalışacak İşletim Sisteminin Seçimi ... 49
7.2 TinyOS İşletim Sisteminin Algılayıcı Düğüme Uyarlanması ... 50
7.2.1 Java 1.5 JDK Yüklenmesi ... 50
7.2.2 Yerel Derleyicilerin Yüklenmesi ... 50
7.2.3 nesC Derleyicisinin Yüklenmesi ... 51
7.2.4 TinyOS 2.x Kaynak Ağacının Yüklenmesi ... 51
7.2.5 Graphviz'in Yüklenmesi ... 51
7.3 TinyOS Üzerinde vf-1a Algılayıcı Düğüm Platformunun Oluşturulması ... 52
7.3.1 .platform Dosyası ... 52
7.3.2 hardware.h Dosyası ... 53
7.3.3 vf-1a İçin make Tanımlama ... 54
7.3.4 PlatformC.nc ve PlatformP.nc Dosyaları ... 55
7.3.5 PlatformLedsC.nc Dosyası ... 55
7.3.6 vf-1a Düğümü Üzerindeki Algılayıcıların Sürücü Dosyaları ... 56
8. İŞLETİM SİSTEMİNİ DÜĞÜM ÜZERİNDE TEST EDEN ÖRNEK UYGULAMA ... 59
9. SONUÇ ... 62
KAYNAKLAR ... 63
ÖZGEÇMİŞ ... 66

SİMGE LİSTESİ

ej	paket işleme süresi
Et	paket işleme görevinin ortalama görev bitirme zamanı
ik	işletim sistemin boş süresi
J	paket sayısı
K	işletim sistemin boş süre sayısı
ls	algılama görevinin süresi
It	işletim sistemin boş süre yüzdesi
T	test işlemi sayısı

KISALTMA LİSTESİ

AM	Active Message
CEC	Cooperative Engagement Capability
DARPA	Defense Advanced Research Agency
DSN	Distributed Sensor Networks
EISLAB	Embedded Internet System Laboratory
EKG	Electrocardiograph
FLL	Frequency Locked Loop
GPS	Global Positioning System (Küresel Yer bulma Sistemi)
HTML	HyperText Markup Language
I2C	Inter Integrated Circuits
IP	Internet Protocol
JTAG	Joint Test Action Group
LED	Light Emitting Diode
MEMS	Micro-elektromechanical Systems
MIT	Massachusetts Institute of Technology
NESL	Networked and Embedded System Laboratory
PLL	Phase Locked Loop
QLP	Quad Leadless Package
SensIT	Sensor Information Technology
SOSUS	Sound Surveillance System
SPI	Serial Peripheral Interface
SPLICE	Signal Processing Language and Interactive Computing Enviroment
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus

ŞEKİL LİSTESİ

Şekil 2.1 DSN projesi kapsamında kullanılan akustik algılayıcılar, algılayıcı düğüm ve ekipmanlar (Chong, Kumar, 2003) 4
Şekil 2.2 Tipik bir telsiz algılayıcı düğüm mimarisi 5
Şekil 2.3 29 Palms denemelerinde kullanılan düğümün paketlenmiş hali[6] 7
Şekil 2.4 29 Palms denemesinde kullanılan insansız hava aracı[6] 7
Şekil 2.5 29 Palms denemesinde takip edilen araçlardan biri (Dragon)[6] 8
Şekil 2.6 Smart Dust projesinde geliştirilen bir prototip[7] 8
Şekil 3.1 Great Duck Island projesinde oluşturulan telsiz algılayıcı ağın yapısı (Mainwaring, Polastre, Szewczyk, Culler, Anderson, 2002) 11
Şekil 3.2 Great Duck Island Projesinde kullanılan koruyucu kılıf içindeki telsiz algılayıcı düğümler (Mainwaring, Polastre, Szewczyk, Culler, Anderson, 2002) 12
Şekil 3.3 Hogthrob projesinde kullanılan düğümün mikrodenetleyici kartı [8] 13
Şekil 3.4 Hogthrob projesinde kullanılan düğümün telsiz iletişim kartı[8] 14
Şekil 3.5 ZebraNet projesi kapsamında geliştirilen prototip(Zhang, Sadler, Lyon, Martonosi, 2004) 15
Şekil 3.6 Pulse oximeter düğümü[10] 16
Şekil 3.7 Smiths Medical PM'in geliştirdiği oksijen ölçüm kartı[10] 17
Şekil 3.8 EKG düğümü [10] 18
Şekil 3.9 Felçli hastaların izlenmesi için geliştirilen düğüm[10] 19
Şekil 4.1 WeC telsiz algılayıcı düğümü[11] 21
Şekil 4.2 Rene telsiz algılayıcı düğümü[11] 21
Şekil 4.3 Rene için hazırlanmış olan algılayıcı kart[11] 22
Şekil 4.4 Dot telsiz algılayıcı düğümü[11] 22
Şekil 4.5 UC Berkeley tarafından geliştirilmiş olan telsiz algılayıcı düğümlerin karşılaştırılması[11] 23
Şekil 4.6 XYZ düğümüne ait çeşitli resimler[16] 24
Şekil 4.7 ENALAB içerisinde oluşturulan telsiz algılayıcı ağın yapısı[16] 25
Şekil 4.8 Mulle düğümünün boyutlarını gösteren şekil[17] 25
Şekil 4.9 Illinois Üniversitesi tarafından geliştirilen düğümün üstten ve alttan görünüşü[18] 26
Şekil 4.10 Eyes projesi kapsamında geliştirilen prototip düğüm[19] 26
Şekil 4.11 tmote invent telsiz algılayıcı düğüm[20] 27
Şekil 4.12 MicaZ düğümleri ve çeşitli algılayıcı kartlar[21] 28
Şekil 5.1 TinyOS işletim sisteminin kaynak yönetimi[22] 29

Şekil 5.2 İletişim bileşeninin grafiksel gösterimi[22] 30
Şekil 5.3 İletişim bileşeninin yazılımsal tanımlaması[22] 31
Şekil 5.4 TinyOS bileşenleri ile oluşturulan örnek bir uygulama[22] 32
Şekil 5.5 MANTIS işletim sisteminin blok diagramı(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005) 33
Şekil 5.6 MANTIS içerisindeki iletişim ve sürücü katmanları(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005) 34
Şekil 5.7 MANTIS işletim sistemi için yazılmış örnek bir uygulama(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005) 35
Şekil 5.8 AVRX için geliştirilen basit bir görev[25] 36
Şekil 5.9 AVRX 2.6 için yazılmış olan örnek uygulama [25] 37
Şekil 5.10 İkili Ağaç(Duffy, Roedig, Herbert, Sreenan, 2007) 38
Şekil 5.11 İkili Ağaç(Duffy, Roedig, Herbert, Sreenan, 2007) 39
Şekil 5.12 Boşta Kalma Zamanı Yüzdesi I_t (Duffy, Roedig, Herbert, Sreenan, 2007) 40
Şekil 6.1 EL7900 iç yapısı ve temel bağlantı biçimi[29] 43
Şekil 6.2 EL7900'ün çıkış geriliminin ışığın parlaklığına göre değişimi[29] 44
Şekil 6.3 EL7900'ün çıkış geriliminin ışığın dalga boyuna göre değişimi[29] 44
Şekil 6.4 SHT15 sıcaklık ve nem ölçüm doğrulukları[27] 46
Şekil 6.5 DS1629 tarih registerlarının yapısı[30] 48

ÇİZELGE LİSTESİ

Çizelge 5.1 Bellek Kullanımları(Duffy, Roedig, Herbert, Sreenan, 2007) 38
Çizelge 6.1 Kullanılan algılayıcıların listesi (Tayşi, 2006) 43
Çizelge 6.2 SHT15'e ait sapma değerleri[27] 45
Çizelge 6.3 SHT15 için nem ve sıcaklık ölçüm özellikleri[27] 45
Çizelge 6.4 SP30 basınç ölçüm aralıkları[28] 47
Çizelge 6.5 SP30 sıcaklık ölçüm aralığı[28] 47
Çizelge 6.6 SP30 ivme ölçüm aralığı[28] 48
Çizelge 7.1 TinyOS 2.x ortam değişkenlerinin tutulduğu alan 51

ÖNSÖZ

Tezimi gerçekleştirme sürecinde, bilgisini ve deneyimlerini bana aktaran değerli hocam ve tez danışmanım Yrd. Doç. Dr. A. Gökhan Yavuz'a, tez çalışmama başlamam ve bitirmem için beni motive eden Arş. Gör. Z. Cihan Tayşi ve Yüksek Bilgisayar Mühendisi Turgut Genç'e ve Y.T.Ü Bilgisayar Mühendisliği Bölümündeki bütün hocalarıma ve arkadaşlarıma çok teşekkür ederim.

Hayatım boyunca desteklerini hiçbir zaman eksik etmeyen, beni yüreklendiren ve bana güvenen sevgili anneme, babama ve kardeşime de ayrıca teşekkür ederim.

ÖZET

İlk olarak askeri uygulamalarda savunma amaçlı kullanılmaya başlanan telsiz algılayıcı ağlar, mikro-elektromekanik sistemler (MEMS) ve telsiz haberleşme sistemlerindeki gelişmeler sonucunda, günümüzde birçok alanda kullanılmaya başlanmıştır. Bu uygulama alanlarından en yaygın olanları; doğal yaşam alanlarının izlenmesi, tarım ürünlerinin gelişimlerinin izlenmesi, endüstriyel kontrol ve takip sistemleri, erken uyarı sistemleri, kapalı ve açık alanların güvenliğinin sağlanmasıdır.

Telsiz algılayıcı ağlar, çevreden bilgi toplayabilen, aldığı bilgiyi bir başka düğüme veya merkeze aktarabilen düğümlerden oluşur. Telsiz algılayıcı düğümler, çevreden bilgi almayı sağlayan çeşitli sayıda ve türde algılayıcılar, algılayıcılardan alınan bilginin saklanması için ikincil depolama birimi, düğümlerin birbirleriyle haberleşmesini sağlayan telsiz haberleşme birimi, enerji ihtiyacının karşılanmasını sağlayan besleme (güç) birimi ve tüm bu işlemlerin gerçekleştirilmesinden sorumlu olan işlemci biriminden oluşmaktadır.

Bu tezde, telsiz algılayıcı ağlarda kullanılmak üzere tasarlanmış varolan bir algılayıcı düğümün üzerine varolan bir işletim sisteminin uyarlanması hedeflenmiştir.

Düğüm üzerinde çalışan işletim sistemi yukarıda belirtilen tipteki telsiz algılayıcı ağ uygulamalarının yazılımsal ihtiyaçlarını karşılayacak niteliktedir. Benzeri uygulamalar için geliştirilen donanımlarda yaygın olarak kullanılan işletim sistemleri incelenmiş ve bu doğrultuda fazla işlemci gücü gerektirmeyen ve az enerji harcayan TinyOS işletim sisteminin mevcut telsiz algılayıcı düğüm üzerinde çalışacak şekilde uyarlanmasına karar verilmiştir. İşletim sistemi, algılayıcı düğüm üzerinde bulunan algılayıcıların yazılımsal olarak aktif veya pasif duruma getirilebilmesini desteklemektedir. Telsiz algılayıcı ağlarda bulunan algılayıcıların kullandıkları temel iletişim protokollerinin kütüphaneleri ve algılayıcı düğüm dahilinde bulunan algılayıcılar için oluşturulacak olan sürücüler işletim sistemi içerisine entegre edilmiştir. Düğümün üzerinde mevcut olan algılayıcıların dışında algılayıcı eklendiğinde bu algılayıcıların sorunsuz çalışabilmesi için, yeni sürücüler oluşturulması ve işletim sistemi içerisine entegre edilmesi için gerekli altyapı hazırlanmıştır.

Uyarlanan işletim sisteminin telsiz algılayıcı düğüm ile uyumluluğunu test etmek amacıyla örnek bir uygulama gerçekleştirilmiştir.

Anahtar kelimeler: İşletim Sistemi, Telsiz Algılayıcı Ağlar, Telsiz Algılayıcı, Algılayıcı Ağlar.

ABSTRACT

As a result of the achievements in Micro-Electro-Mechanical Systems (MEMS) and wireless communication systems, wireless sensor networks, which were firstly used for military defense applications, are recently starting to be used almost in any area. The most commonly used applications of wireless sensor networks are tracking of natural life, intelligent agriculture, industrial control and tracking systems, early alert systems and systems for security of open and closed areas.

Wireless sensor networks consist of nodes which are capable of collecting data from its envoriment and deliver that data to another node or central node. Wireless sensor nodes consist of a sensor module to gather information from the envoriment, a secondary storage module to store data that is collected from the sensors, a commucation module to communicate with the other nodes, a power management module and a microcontroller to manage all operations.

The aim of this project is to port an existing operating system to an existing general purpose wireless sensor node.

The operating system of the node fulfills the software requirements of the above mentioned applications. Operating systems of existing wireless sensor nodes are analyzed and it is decided to port the TinyOS operating system which will not require high processing power and is energy-efficient to an existing wireless sensor node. The operating system supports the activation or deactivation of the sensors which are on the sensor node at the software level. Common communication protocols used by sensors and the corresponding drivers for the sensors on the sensor node are integrated into the operating system. For addition of sensors which are not on the sensor node, the required drivers of the sensors can be implemented and integrated into the operating system by using the provided development platform.

A test application is developed to test the compatibility of the operating system with the wireless sensor node.

Keywords: Operating Systems, Wireless Sensor Networks, Sensor Node, Wireless Sensor.

1. GİRİŞ

Son yıllarda telsiz iletişim ve mikro-elektromekanik sistem (MEMS) teknolojilerindeki hızlı gelişmeler, düşük maliyetli, düşük güç tüketimine sahip, çok işlevli ve küçük boyutlu telsiz algılayıcılardan oluşan telsiz algılayıcı ağların (wireless sensor networks) geliştirilmesine olanak sağlamıştır. Telsiz algılayıcı ağlar ilk olarak askeri amaçlı savunma uygulamalarında kullanılmıştır. Günümüzde ise telsiz algılayıcı ağların kullanım alanları arasında doğal yaşam alanlarının izlenmesi, tarım ürünlerinin gelişimlerinin izlenmesi, endüstriyel kontrol ve izleme sistemleri, ev otomasyon sistemleri, güvenlik uygulamaları ve tedarik zinciri uygulamaları bulunmaktadır.

Bu tez kapsamında daha önceden varolan TinyOS işletim sistemi yukarıda belirtilmiş olan uygulamalarda kullanılabilecek şekilde tasarlanmış genel amaçlı bir telsiz algılayıcı düğümün üzerinde çalışacak şekilde uyarlanmıştır. İşletim sisteminin uyarlanması sırasında; kablosuz algılayıcı düğümlerin daha uzun süre çalışabilmeleri için dikkat edilmesi gereken enerji kıstasları gözönünde bulundurulmuştur. Bunun için, işletim sisteminin az işlemci gücü ile çalışması ve düğüm üzerinde bulunan fakat kullanılmayan algılayıcıların istenildiği zaman tekrar aktif hale getirilebilecek şekilde pasif duruma getirilmesi sağlanmıştır. İşletim sisteminin düğüm üzerinde sorunsuz çalışabilmesi için düğümde bulunan algılayıcıların sürücüleri işletim sistemi içerisine entegre edilmiştir. Düğüm üzerinde bulunmayan algılayıcıların sonradan düğüme eklenmesi durumu düşünülerek yeni sürücülerin oluşturulması ve işletim sistemine entegre edilmesi için gereken altyapı hazırlanmıştır.

Tez çalışması kapsamında öncelikle uyarlanacak işletim sisteminin üzerinde çalışacağı düğümde bulunan donanım elemanları hakkında bilgi edinilmiştir. Sonraki adımda ise bugüne kadar geliştirilmiş varolan işletim sistemleri incelenmiştir. Elde edilen bu bilgiler ışığında donanımla uyumlu olarak hizmet verecek olan TinyOS işletim sisteminin algılayıcı düğüme uyarlanmasına karar verilmiştir. İşletim sisteminin donanımla uyumluluğunun test edilmesi için örnek bir uygulama geliştirilmiştir. Örnek uygulamanın çalıştırılması ve incelenmesi sonucunda alınan geri-besleme sonucunda işletim sistemi üzerinde gerekli değişiklikler ve güncellemeler yapılmıştır.

Tezin 2. bölümünde telsiz algılayıcı ağların gelişimi ve bu alandaki mevcut teknolojik gelişmeler, 3. bölümde bugüne kadar oluşturulmuş olan telsiz algılayıcı ağlar, 4. bölümde bu ağların yapısında ve diğer uygulamalarda kullanılmak üzere üretilmiş olan telsiz algılayıcı düğümler incelenmiştir. 5. bölümde telsiz algılayıcı düğümler için tasarlanmış olan işletim

sistemleri incelenmiştir. 6. bölümde, yapılan incelemeler sonucunda uyarlanmış işletim sisteminin üzerinde çalışacağı düğümün donanım elemanları hakkında bilgi verilmiştir. 7. bölümde düğüm üzerinde çalışacak şekilde uyarlanacak işletim sisteminin seçimi ve düğüm üzerine uyarlanması anlatılmıştır. 8. bölümde ise geliştirilen işletim sisteminin donanımla uyumluluk testi için geliştirilen örnek uygulama hakkında bilgi verilmiştir.

2. TELSİZ ALGILAYICI AĞLARIN GELİŞİMİ

Telsiz algılayıcı ağların temelleri dağıtık algılayıcı ağlara (Distributed Sensor Networks) dayanmaktadır. Dağıtık algılayıcı ağların ilk uygulamalarından biri 1950'li yıllarda kurulumuna bağlanmış olan Sound Surveillance System (SOSUS)'tur[1]. SOSUS, soğuk savaş sırasında Amerikan donanması tarafından, Rus denizaltılarının izlenmesi için Atlantik ve Pasifik okyanuslarına kurulmuş ve kullanılmıştır. SOSUS'un yapısında okyanus dibine yerleştirilmiş olan akustik algılayıcı (hydrophones) dizileri bulunmaktadır. Bu algılayıcı dizileri okyanus dibine döşenmiş iletişim kabloları ile kıyılardaki merkezlere bağlanmıştır. SOSUS sisteminin geliştirilmesi ve başarılı şekilde denizaltıları takip etmeye başlaması, çok daha sessiz motorlara sahip denizaltıların geliştirilme sürecini başlatmıştır. Çok daha sessiz denizaltıların geliştirilmesi de daha hassas akustik algılayıcıların geliştirilmesine ve SOSUS'un yeni bir kıyı güvenlik sistemi ile değiştirilmesine neden olmuştur. Günümüzde SOSUS okyanus dibindeki titreşimler ve okyanus canlılarının izlenmesi için aktif olarak kullanılmaktadır.(Chong, Kumar, 2003)

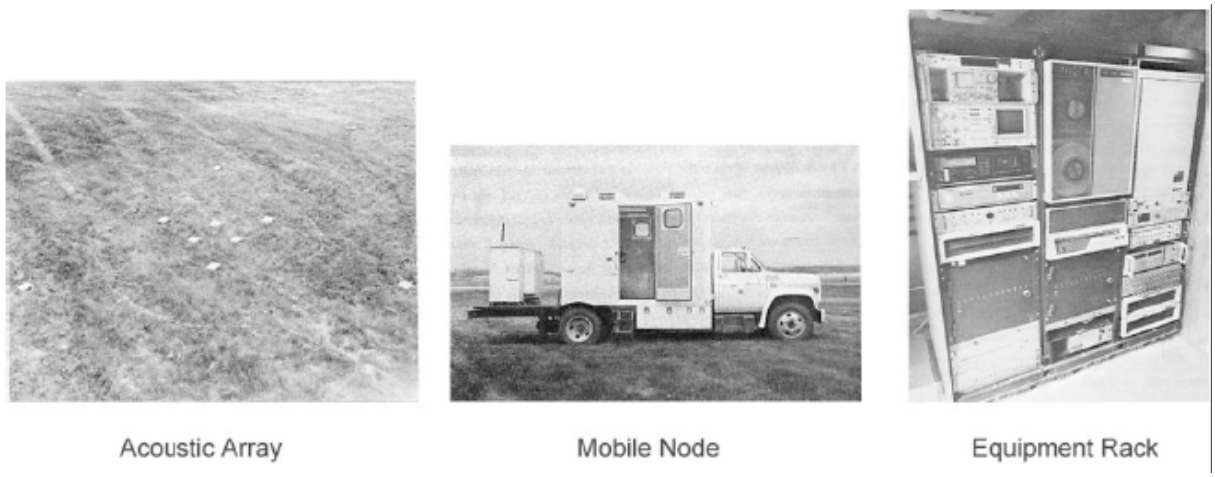
Dağıtık algılayıcı ağların teknolojik bileşenleri ilk kez 1978 yılında gerçekleştirilen Distributed Sensors Net seminerinde tanımlanmıştır. Bu bileşenler akustik algılayıcılar, kaynak paylaşımlı bir ağ üzerinde çalışan işlemleri birleştiren üst seviye iletişim protokolleri, bilgi işleme teknikleri, yer bulma (node localization) algoritmaları ve dağıtık sistemler üzerinde çalışan dinamik olarak değiştirilebilen dağıtık yazılımlardır.(Chong, Kumar, 2003)

Dağıtık algılayıcı ağlar üzerindeki ilk bilimsel çalışmalar ise 1980'li yıllarda Defense Advanced Research Agency (DARPA) tarafından başlatılan Distributed Sensor Networks (DSN) programı kapsamında başlamıştır. Program kapsamında dağıtık akustik izleme problemi örnek uygulama olarak seçilmiştir. Bu uygulamanın farklı kısımları farklı üniversiteler ve kuruluşlar tarafından geliştirilmiştir.(Chong, Kumar, 2003)

Proje kapsamında Carnegie Mellon Üniversitesi'nde, algılayıcı bir ağ üzerinde dağıtık olarak bulunan kaynaklara hata hoşgörülü (fault-tolerant) ve şeffaf (transparent) şekilde erişimi sağlayan, iletişim-merkezli (communication-oriented) Accent adında bir işletim sistemi geliştirilmiştir. Daha sonra bu işletim sistemi geliştirilerek ticari olarak kullanılan Mach işletim sistemi oluşturulmuştur.(Chong, Kumar, 2003)

Proje kapsamında hava araçlarının takip edilmesi için kullanılacak olan sinyal işleme teknikleri ise Massachusetts Institute of Technology (MIT) tarafından geliştirilmiştir. Geliştirilen bu tekniklerde insanların gerçek hayatta sinyalleri işleme ve yorumlama

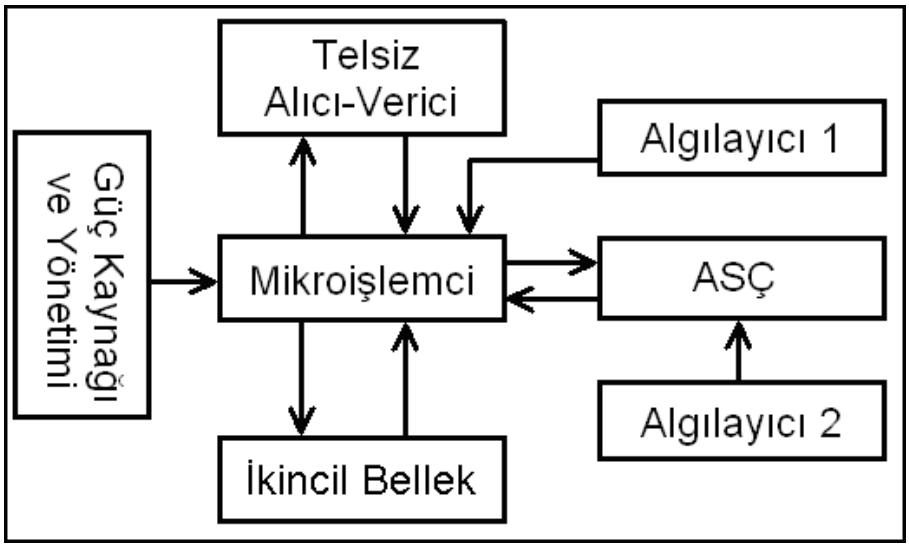
yöntemleri temel alınmıştır. Ayrıca MIT tarafından proje kapsamında dağıtık algılayıcı ağlarda veri analizi ve algoritma geliştirilmesi için kullanılmak üzere, Signal Processing Language and Interactive Computing Environment (SPLICE) sistemi geliştirilmiştir.



Şekil 2.1 DSN projesi kapsamında kullanılan akustik algılayıcılar, algılayıcı düğüm ve ekipmanlar (Chong, Kumar, 2003)

Geliştirilen projenin test edilmesi amacıyla MIT Lincoln Laboratuarı tarafından bir test ortamı oluşturulmuştur. Bu test ortamında, ortamdan bilgi toplanması için 9 adet akustik algılayıcı ve bu algılayıcılardan gelen bilginin işlenmesi için PDP11/34 işlemcili bir bilgisayar kullanılmıştır. Hedef takibi (target tracking) için ise 256 kB bellek birimine sahip 3 adet MC68000 işlemci ve 512 kB bellek birimine sahip bir bilgisayar kullanılmıştır. Şekil 2.1'de en solda test sırasında kullanılan 9 adet akustik algılayıcı dizisi görülmektedir. Bu algılayıcılar ortak merkezli 3 adet üçgen şeklinde yerleştirilmiştir. Orta resimde hareketli düğüm, sağdaki resimde ise düğüm içerisine yerleştirilmiş olan bilgisayar sistemleri görülmektedir. Hareketli düğümün elektrik ihtiyacı kamyonun arka tarafında yer alan sessiz jeneratörler ile sağlanmaktadır. Oluşturulan bu sistem, boyut ve performans açısından 1980'li yıllar için en son teknoloji ürünüdür. Yapılan test uçuşlarında oluşturulan akustik algılayıcı ağ, hava araçlarını başarılı bir şekilde takip etmiştir. (Chong, Kumar, 2003)

Telsiz algılayıcı düğüm (wireless sensor node); telsiz algılayıcı ağlarda kullanılan, işlem yapma yetisine sahip, algılayıcı bilgisini toplayıp ağ üzerindeki diğer düğümlerle iletişim kurabilen bir donanımdır. Algılayıcı düğümün tipik mimarisi Şekil 2.2'de gösterilmiştir.



Şekil 2.2 Tipik bir telsiz algılayıcı düğüm mimarisi

Her ne kadar dağıtık algılayıcı ağların yapısında çok sayıda ufak boyutta algılayıcı düğümler kullanılması planlanmış olsa da 1980'li ve 1990'lı yıllarda bu özelliklerde bir algılayıcı düğümün geliştirilmesi, gerekli teknolojilerin hazır olmaması nedeniyle mümkün olmamıştır. Ancak buna rağmen özellikle algılayıcı ağların, ağ merkezli savaşlar (network centric warfare) gibi askeri uygulamalarda, oldukça etkili çözümler sunduğu görülmüş ve bu alanlarda çeşitli projeler geliştirilmiştir. Bu projelerden biri de Amerikan donanması tarafından geliştirilen Cooperative Engagement Capability (CEC)'dir.[5] Projenin amacı savaş sistemlerinin, üzerlerindeki çeşitli algılayıcılar vasıtasıyla topladıkları bilgileri birbirleri ile paylaşarak tek bir sistem gibi çalışmasının sağlanmasıdır. Bu yapı temel alınarak hava, kara ve sualtı savunma sistemleri geliştirilmiştir.

2000'li yıllarda özellikle micro-elektromekanik sistemler, telsiz iletişim ve düşük güç tüketimli mikrodenetleyicilerde yaşanan gelişmeler, telsiz algılayıcı ağların ilk düşünüldüğü gibi çok sayıda ufak algılayıcı düğümden oluşması fikrini uygulanabilir hale getirmiştir. Yaşanan teknolojik gelişmeler algılayıcı düğümlerin küçülmesini sağlamanın yanı sıra maaliyetlerinin de düşmesine neden olmuştur. Böylece 1990'lı yılların sonlarına kadar çok büyük bütçeli askeri güvenlik uygulamarında kullanılan ve oldukça maaliyetli olan telsiz algılayıcı ağlar, sivil araştırmacılar tarafından birçok uygulama alanında kullanılmaya başlanmıştır.

Bu uygulama alanlarından en yaygın olanları, doğal yaşam alanlarının izlenmesi, tarım ürünlerinin gelişimlerinin izlenmesi, endüstriyel kontrol ve izleme sistemleri, erken uyarı sistemleri, kapalı ve açık alanların güvenliğinin sağlanmasıdır.

Telsiz algılayıcı ağlar, özellikle insan etkisinin en aza indirilmek istenildiği doğal yaşam alanlarının izlenmesi uygulamalarına çok uygundur. Great Duck Island[2] ve ZebraNet[3] projeleri bu alanda yapılmış uygulamarın en bilinen örnekleridir. İnsan gücüyle yapılması oldukça zor ve maliyetli olan tarım ürünlerinin gelişimlerinin izlenmesi işlemi, telsiz algılayıcı ağlar ile oldukça ucuz ve kolay gerçekleştirilmektedir. Her gün belirli aralıklarla ortamla ilgili nem, sıcaklık vb. bilgileri bir merkeze gönderecek telsiz algılayıcı düğümler ürün yetiştiricisinin ürünlerinin gelişimiyle ilgili oldukça detaylı bilgiye sahip olmasını sağlayacaktır. Endüstriyel kontrol ve izleme alanında telsiz algılayıcı ağların kullanımına ilişkin en iyi örnek Hogthrob[8] projesi olacaktır. Danimarka'da domuz yetiştiricileri birliğinin (Commitee of Pig Production) kontrolünde çeşitli kurumların katılımıyla gerçekleştirilen bu projede hedeflenen domuzların çiftleşme ve hamilelik dönemlerinin izlenmesi ve böylece üretimin arttırılmasıdır.

Telsiz algılayıcı ağların kullanım alanlarının artmasıyla birlikte, bu ağların yapısında kullanılan düğümlerin üzerinde çalışacak yazılımların geliştirilmesi ihtiyacı da ortaya çıkmıştır. Bu ihtiyacı karşılamak amacıyla çeşitli kurumlar tarafından değişik işletim sistemleri ve yazılımlar geliştirilmiştir. Günümüzde bu yazılımlar arasında en çok kullanılanı UC Berkeley tarafından geliştirilen TinyOS'tur[22]. Bunun dışında MANTIS[23] ve SOS[24] gibi işletim sistemleri ve düğümlere özel olarak geliştirilmiş yazılımlar da mevcuttur. Tüm bu yazılımların ortak özelliği güç tüketimini en aza indirerek düğümün çalışma ömrünü uzatmaktır.

Algılayıcılar, telsiz iletişim birimleri ve düşük güç tüketimli mikrodenetleyicilerde yaşanan gelişmeler sivil telsiz algılayıcı ağ uygulamalarının geliştirilmesinin yanı sıra askeri uygulamaların yeniden gözden geçirilmesine ve ilk başta planlandığı şekliyle yeniden oluşturulmalarına neden olmuştur. DARPA tarafından 1980'lerde geliştirilen hava araçlarının izlenmesi projesi tekrar gözden geçirilerek, hem hava hem de kara araçlarının izlenmesinde kullanılmak üzere telsiz algılayıcı ağların tasarlanması amacıyla Sensor Information Technology (SensIT)[4] programı başlatılmıştır. Program kapsamında çeşitli telsiz algılayıcı düğümler geliştirilmiş ve geliştirilen bu düğümlerden oluşturulan ağlar üzerinde denemeler yapılmıştır.

Şekil 2.3 29 Palms denemelerinde kullanılan düğümün paketlenmiş hali[6]

Şekil 2.3'te UC Berkeley tarafından 29 Palms California'da gerçekleştirilen bu denemelerden birinde kullanılan telsiz algılayıcı düğümü ve bu düğümün paketlenmiş hali görülmektedir. Bu denemede algılayıcı düğümlerin, insansız bir uçaktan atılarak izlenecek olan yol ve çevresine yerleştirilmesi, yerleştirilen düğümler arasında eş zamansalı bir ağın oluşturulması, ağın yakınlarından geçen araçların tespit ve takip edilmesi, izlenen araçların bilgilerinin insansız hava aracına aktarılması ve hava aracının araçlarlarla ilgili bilgiyi bir yer merkezine aktarması hedeflenmektedir.

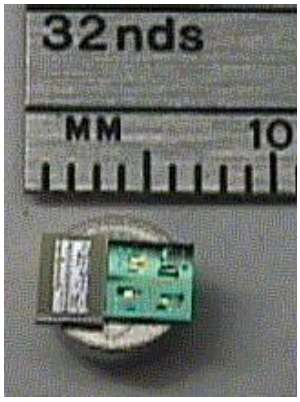
Şekil 2.4'te kullanılan insansız hava aracı, Şekil 2.5'te ise izlenen araçlardan biri görülmektedir. Yapılan 3 günlük testler sonucunda sistemin planlandığı gibi çalışması sağlanmıştır[6].

Şekil 2.4 29 Palms denemesinde kullanılan insansız hava aracı[6]

Testlerde UC Berkeley Üniversitesi tarafından geliştirilen Rene adlı telsiz algılayıcı düğüm kullanılmıştır. Düğüm üzerine yakından geçen araçların tespit edilebilmesi için 2 boyutlu manyetik alan ölçen (2 axis magnetometer) bir algılayıcı yerleştirilmiştir. Düğümler üzerinde yine UC Berkeley Üniversitesi tarafından geliştirlmiş olan TinyOS işletim sistemi çalıştırılmıştır.

Şekil 2.5 29 Palms denemesinde takip edilen araçlardan biri (Dragon)[6]

Günümüzde geliştirilen telsiz algılayıcı düğümlerin boyutları artık milimetrelerle ölçülmektedir. Ancak hedeflenen çok daha küçük boyutlardaki düğümlerin oluşturulmasıdır. Bu kapsamda başlatılan projelerden biri olan Smart Dust kapsamında geliştirilen prototiplerden biri Şekil 2.6'da görülmektedir. Prototip'in yapısında 1 adet optik iletişim birimi, 1 adet mikrodenetleyici ve işitme cihazlarında kullanılan türde bir pil birimi mevcuttur. Prototip, 0.25 mikron teknolojisi ile geliştirilmiştir.



Şekil 2.6 Smart Dust projesinde geliştirilen bir prototip[7]

Düğümlerin boyutlarının küçülmesi, düğümlerin güç ihtiyaçlarının karşılanmasında kullanılan pillerinde küçük boyutlarda olmasını zorunlu kılmaktadır. Bu yüzden düğümlerin kesintisiz çalışma zamanları birkaç gün ile sınırlıdır. Düğümlerin çalışma zamanlarını uzatabilmek için, çeşitli yazılımsal yöntemler geliştirilmiştir. Bu yöntemlerden en yaygın olanı ise düğümlerin belirli zaman aralıklarında çevreden bilgi toplamaları ve bu zamanlar dışında kendilerini kapatmalarıdır. Burada zaman aralığı ne kadar uzun seçilirse düğümün çalışma ömrü o kadar uzun olacaktır, diğer yandan zaman aralığının uzaması etraftan toplanmış olan bilginin hassasiyetini azaltacaktır. Düğümlerin çalışma ömürlerinin arttırılması için daha az güç tüketimine sahip mikrodenetleyicilerin, küçük ancak çok daha güçlü pil birimlerinin ve "wake-up radio" teknolojisini kullanan telsiz iletişim birimlerinin geliştirilmesine ihtiyaç duyulmaktadır.

3. KULLANIMDA OLAN TELSİZ ALGILAYICI AĞLAR

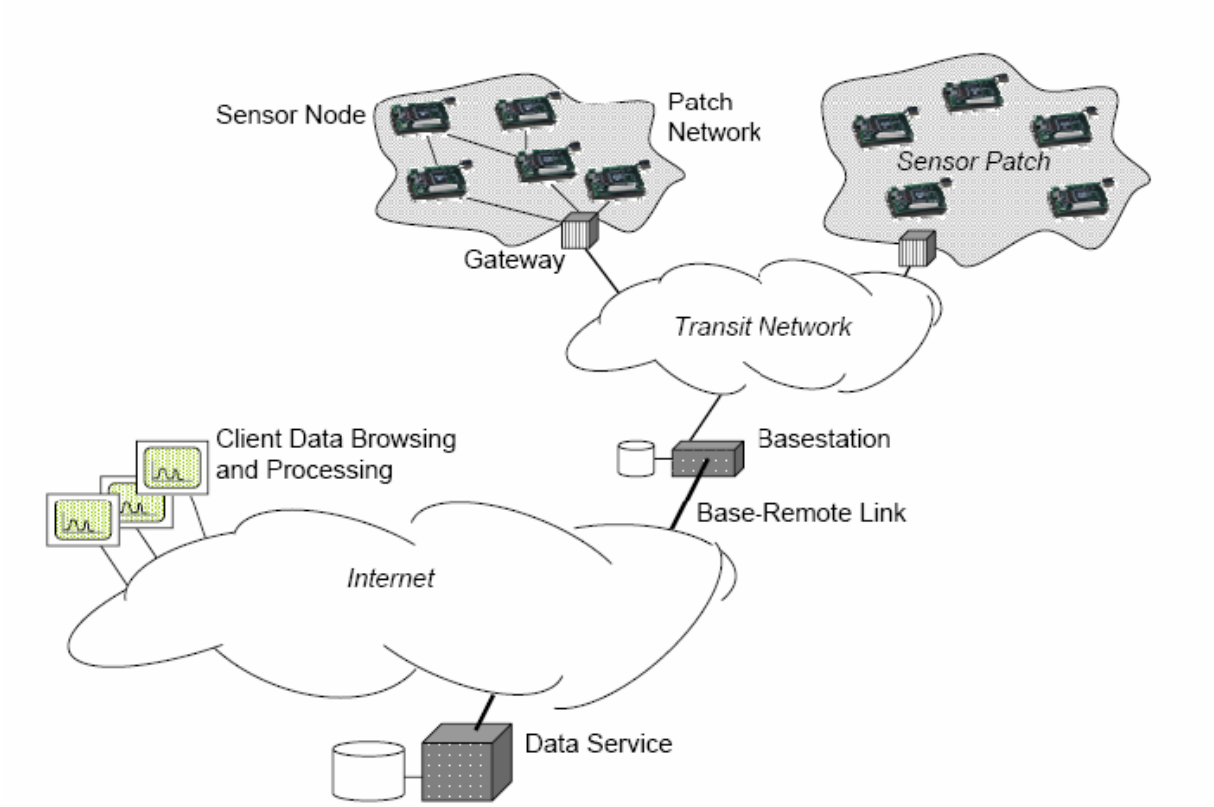
Günümüzde telsiz algılayıcı ağların kullanıldığı pek çok uygulama alanı mevcuttur. Bu uygulama alanlarından önde gelenleri ise; telsiz algılayıcıların, insan müdahelesini dolayısıyla ortam üzerindeki etkisini en aza indirdiği doğal hayatın izlenmesi, kablolama zorluklarının olduğu endüstriyel kontrol ve izleme uygulamalarıdır. Bu bölüm kapsamında, bugüne kadar oluşturulmuş doğal hayatın gözlemlenmesinde ve endüstriyel uygulamalarda kullanılmış olan telsiz algılayıcı ağlardan en önemlileri incelenmiş ve özellikleri detaylı bir şekilde ortaya konmuştur.

3.1 Great Duck Island

2002 yılının ilk baharında Berkeley'deki Intel araştırma laboratuarı, California Üniversitesi ve Atlantik Kolejinin işbirliği ile doğal hayatın ve hayvanların yaşam alanlarının insan etkileşimi olmaksızın gözlenmesinde kullanılmak üzere doğal ortam izleme teçhizatının oluşturulmasını amaçlayan bir program başlatılmıştır[2]. Program kapsamında pilot bölge olarak Amerika'nın Maine eyaletindeki Great Duck adası seçilmiş ve ilk etapta adadaki kuşların yuvalarını yakınlarına 32 adet telsiz algılayıcı düğüm yerleştirilmiştir.

Kullanılan telsiz algılayıcı düğümler, bir mikrodenetleyici birimi, bir adet düşük enerji ihtiyaçlı radyo birimi, bir adet ikincil bellek birimi, piller ve algılayıcı birimlerinden oluşmaktadır. Sistemde ısı, nem, hava basıncı ve orta menzilli kızıl ötesi algılayıcılar bulunmaktadır.

Adaya yerleştirilmiş olan telsiz algılayıcı düğümler, belirli aralıklarla üzerlerindeki algılayıcılardan gelen bilgileri okur ve bu bilgileri adadaki bilgisayar merkezlerine aktarır. Bu merkezlerde toplanan bilgiler de Internet üzerinden yayınlanmaktadır. Şekil 3.1'de de görüldüğü gibi çok katmanlı bir ağ oluşturulmuştur. Oluşturulan ağın en uç noktasında telsiz algılayıcı düğümler bulunmaktadır. Bu düğümler topladıkları bilgileri kendilerine yakın olan merkezlere iletirler. Ara merkezlerde toplanan bilgiler ana merkeze aktarılır ve buradan Internet aracılığıyla tüm dünyaya sunulur.



Şekil 3.1 Great Duck Island projesinde oluşturulan telsiz algılayıcı ağın yapısı (Mainwaring, Polastre, Szewczyk, Culler, Anderson, 2002)

Projenin başlamasından 2002 yılının kasım ayına kadar yerleştirilmiş olan 32 adet telsiz algılayıcı düğümden toplam bir milyondan fazla okuma işlemi yapılmıştır. 2003 yılının haziran ayında adadaki ağ, 56 düğümden oluşan yeni bir ağ ile değiştirilmiştir. Yeni oluşturulan bu ağa temmuz ayında 49 adet, ağustos ayında ise 60 adet düğüm daha eklenmiştir. Ağustos ayında ayrıca hava gözlemleme de kullanılmak üzere 25 adet yeni düğüm eklenmiştir. Şekil 3.2'de koruyucu kılıflar içerisinde kullanılan telsiz algılayıcı düğümler görülmektedir. Ağ içerisinde kullanılan telsiz algılayıcı düğümlerin çiğ, yağmur, sel, doğal hava koşullarına ve darbelere karşı dayanıklı olması gerekmektedir.

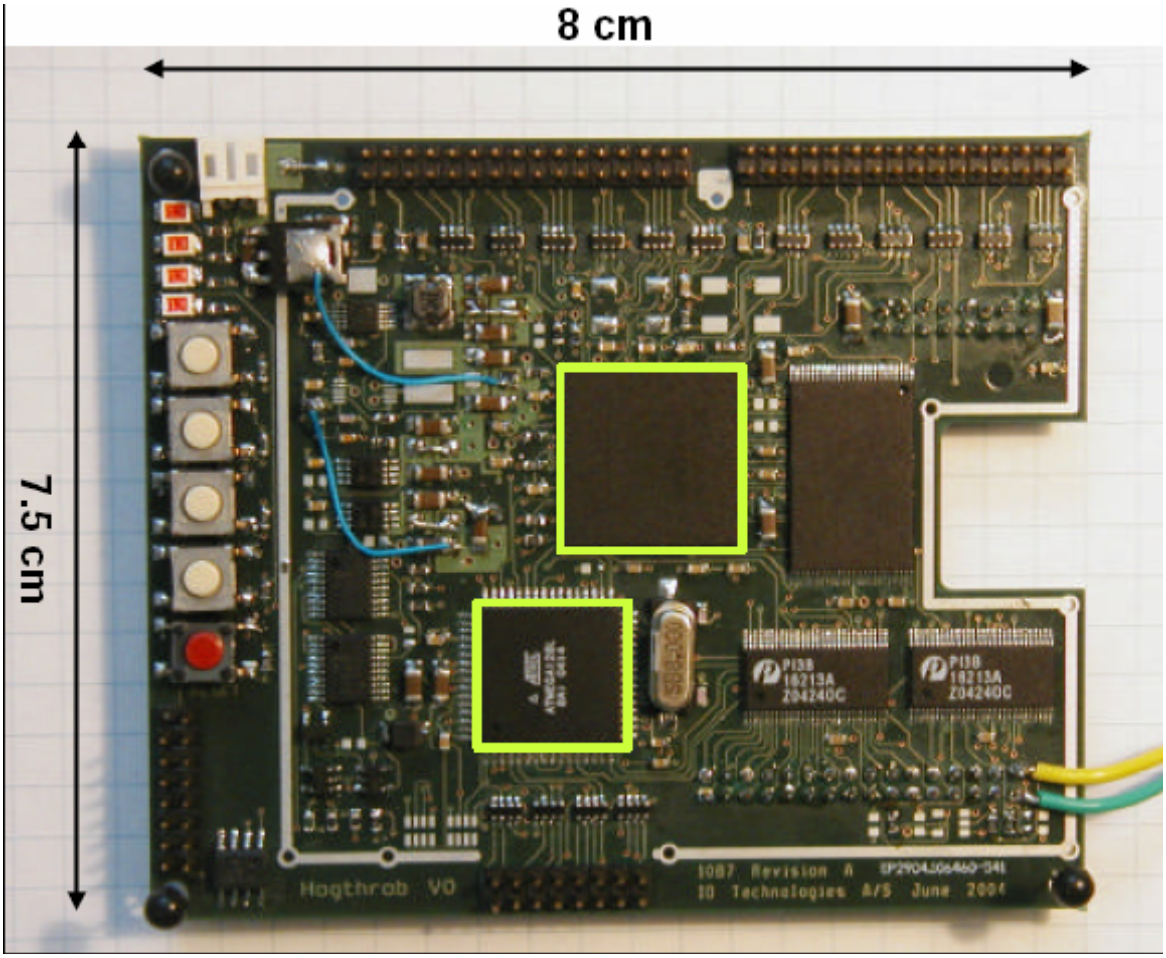
Şekil 3.2 Great Duck Island Projesinde kullanılan koruyucu kılıf içindeki telsiz algılayıcı düğümler (Mainwaring, Polastre, Szewczyk, Culler, Anderson, 2002)

Proje kapsamında Berkeley üniversitesi tarafından geliştirilmiş olan Mica Mote adlı telsiz algılayıcı düğüm kullanılmıştır. Mica Mote dört ana birimden oluşmaktadır. Bu birimler, AtMega 128L[9] mikrodenetleyicinin kullanıldığı işlemci birimi, TR1000[14]'nin kullanıldığı radyo iletişim birimi, AT45DB041B[13]'nin kullanıldığı depolama birimi ve 2 adet AA boyutunda bataryanın kullanıldığı güç birimidir. Düğüme istenilen algılayıcıların bağlanmasını sağlayan 51 pinlik bir ara bağlantı birimi bulunmaktadır. Bu ara bağlantı üzerinden analog, I2C (Inter Integrated Circuits), SPI (Serial Peripheral Interface) ve UART (Universal Asynchronous Receiver/Transmitter) tipinde bağlantılar yapılabilmektedir. Düğüm ve düğümde kullanılan algılayıcıların yapısı ile ilgili detaylı bilgi bölümün sonunda yer almaktadır.

3.2 Hogthrob

Danimarka' da domuz yetiştirilmesinde verimi arttırmak için oluşturulan bu telsiz algılayıcı ağ, DIKU (Department of Computer Science), DTU (Denmarks Tekniske Universitet), KVL (The Royal Veterinary and Agriculture University) ve Danish Comittee for Pig Production tarafından geliştirilmiştir.[8]

2004 yılında başlatılan projedeki amaç domuzların çiftleşme dönemlerinin tespit edilerek üretimin arttırılmasının sağlanmasıdır. Domuzların farklı dönemlerde farklı kafeslerde ve koşullarda tutulması gerekmektedir. Örneğin gebe domuzlar yaklaşık $300m^2$ lik bir kafeste 114 gün süreyle tutulmalıdır. Doğum yapmak üzere olan domuzlar ise 3 ila 4 hafta arasında $4m^2$ lik kafeslerde tek başlarına olmalıdılar. Çiftleşme döneminde ise domuzlar 2 ila 3 hafta arasında onarlı gruplar halinde $50m^2$ lik kafeslerde bekletilirler.



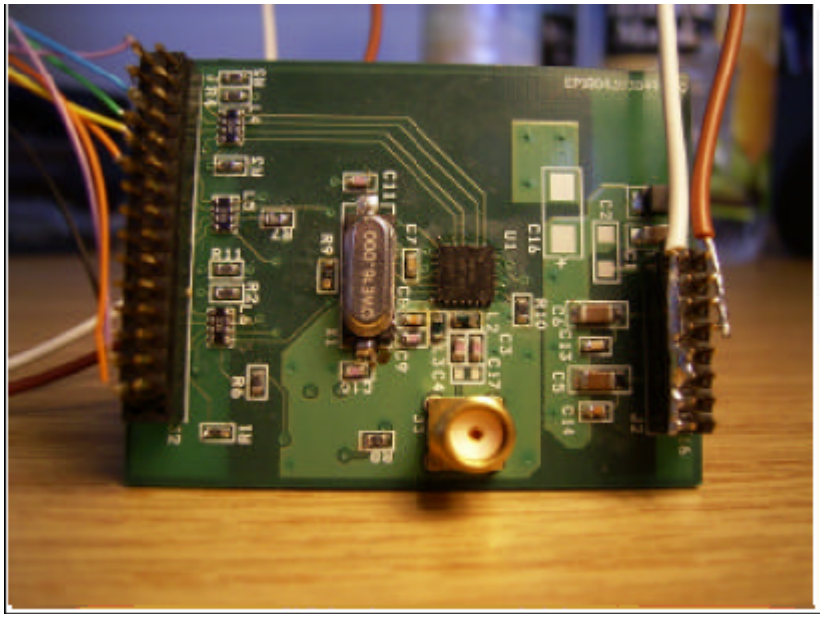
Şekil 3.3 Hogthrob projesinde kullanılan düğümün mikrodenetleyici kartı [8]

Domuzların yetiştirilmesinde kullanılan diğer bir yöntem ise domuzların kulaklarına takılan RFID birimlerinden faydalanılmasıdır. Ancak bu yöntemde bilgilerin okunabilmesi için domuzlara çok yakın olunması (yaklaşık 10m) gerekmektedir. Oysa telsiz algılayıcı düğümlerin kullanılması, domuzlarla fiziksel temasın en aza indirildiği, çiftleşme dönemleri dışında domuzlarla ilgili diğer bilgilere de ulaşılmasını sağlayan çok daha etkili bir çözümdür. Telsiz algılayıcı düğümlerin kullanımı, domuzların çiftleşme dönemlerinin belirlenmesi yanında, domuzların takip edilmesini ve hastalık vb. anormal davranışlarının takibini de sağlamaktadır.

Oluşturulan telsiz algılayıcı ağın oldukça basit bir yapısı vardır. Domuzların üzerlerine yerleştirilmiş olan düğümler bir merkeze, belirli aralıklarla algılayıcıları ile topladıkları bilgileri aktarmaktadır.

Proje kapsamında kullanılan telsiz algılayıcı düğüm Şekil 3.3'te görülmektedir. Algılayıcı

düğümün işlemci birimini Atmega 128L[9] mikro denetleyici oluşturmaktadır. Düğüm üzerinde algılayıcılar aracılığıyla toplanan bilginin saklanması için bir saklama birimi bulunmaktadır. Algılayıcıların bulunduğu birimin ve radyo biriminin bağlanması için algılayıcı düğüm üzerinde ayrı bağlantı yolları bulunmaktadır. Şekil 3.4'te kullanılan radyo modülü görülmektedir.



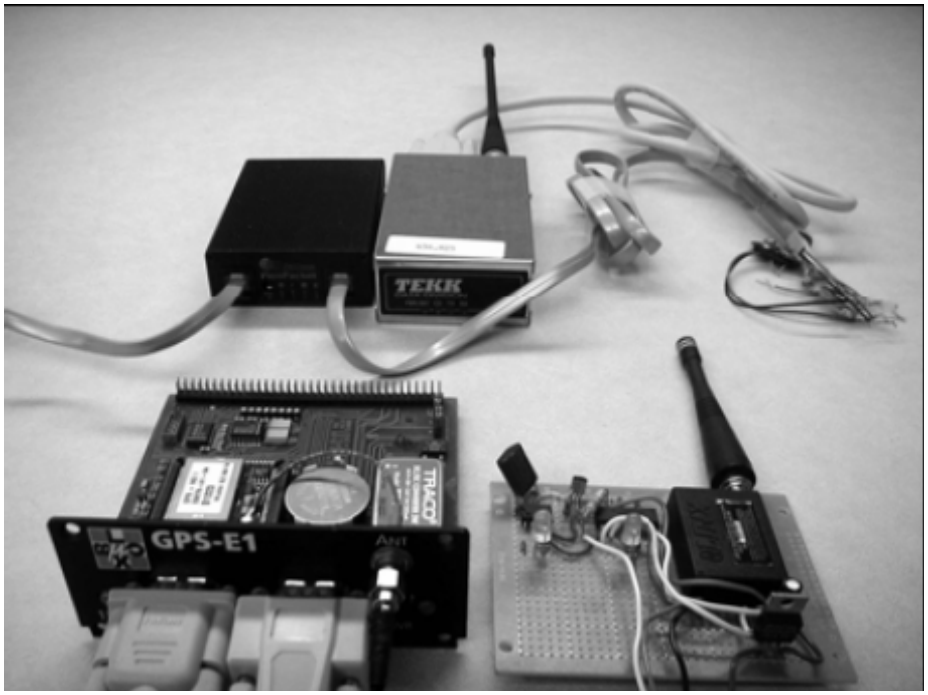
Şekil 3.4 Hogthrob projesinde kullanılan düğümün telsiz iletişim kartı[8]

3.3 ZebraNET

2001 yılında başlatılan bu projedeki amaç, türlerin hem kendi içlerindeki, hem de aralarındaki etkileşimleri ile insanlığın gelişiminin hayvanların doğal yaşamlarına olan etkisini araştırmaktır. Oluşturulan ağ, ismini ilk kez üzerlerinde denendiği zebralardan alsa da, asıl amaçlanan tüm vahşi hayvanların izlenmesine olanak sağlayacak bir yapının oluşturulmasıdır.

Bu izleme işleminin gerçekleştirilebilmesi için öncelikle havyanların yerlerinin bilinmesi gerekmektedir. Bunun için de her düğüm üzerinde bir adet küresel yer bulma sitemi (Global Positioning System - GPS) bulunmaktadır. Ayrıca her düğüm üzerinde hayvanın yeme, içme veya uyuma gibi basit yaşamsal faaliyetleri gerçekleştirdiğinin anlaşılmasını sağlayan algılayıcılar bulunmaktadır. Geliştirilen telsiz algılayıcı düğümlerin herbiri bir boyunluk yardımı ile bir zebranın boynuna yerleştirilmiştir. Oluşturulan yapı içerisinde bilginin toplandığı sabit bir merkez yoktur. Sabit merkezler yerine devamlı yer değiştiren ve gözlemciler tarafından kullanılan merkezler oluşturulmuştur. Bilginin düğümlerden bu

merkeze aktarılması direk veya dolaylı olarak yapılabilmektedir. Her düğüm belirli aralıklarla çevresinde başka düğümler arar ve bilgilerini bu düğümlere aktarır. Bu şekilde her bilginin birden çok kopyasının tutulduğu yedeklemeli bir yapı oluşturulur. Düğümler bir merkez ile iletişime geçtiklerinde ise hem kendisine ait hem de diğer düğümlerden kendisine aktarılmış olan bilgileri merkeze aktarır. Kullanılan boyunlukların, zebralar bayıltıldıktan sonra takıldığı ve bu işlemin oldukça zor ve maaliyetli olduğu düşünülürse kullanılan düğümlerin uzun süre çalışması gerekliliği ortaya çıkmaktadır. Bu yüzden düğümler güneş enerjisini kullanarak düğümün pilinin doldurulmasını sağlayacak bir güç yönetim birimine de sahiptirler.



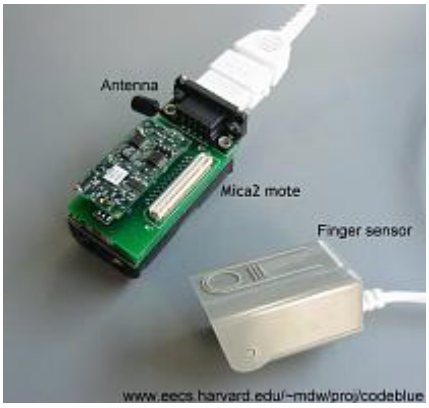
Şekil 3.5 ZebraNet projesi kapsamında geliştirilen prototip(Zhang, Sadler, Lyon, Martonosi, 2004)

Proje kapsamında geliştirilen telsiz algılayıcı düğümün ilk örneklerinden biri Şekil 3.5'te görülmektedir. Düğümlerin tasarımı son halini alana kadar çok çeşitli testlerden geçirilmiştir. Öncelikle New Jersey de evcil atlara yerleştirilen düğümler, daha sonra Amerika'nın Atlantik kıyısındaki adalarda vahşi atlar üzerinde denenmiştir. Bu denemelerden sonra son haline getirlen düğüm 2004 yılının son baharında Kenya'daki Sweetwaters koruma bölgesinde 50 adet zebra üzerine yerleştirilmiş ve 1 ay süreyle denenmiş ve böylece kurulacak olan yapının nasıl olması gerektiği ile ilgili bilgiler toplanmıştır. (Zhang, Sadler, Lyon, Martonosi, 2004)

3.4 Code Blue

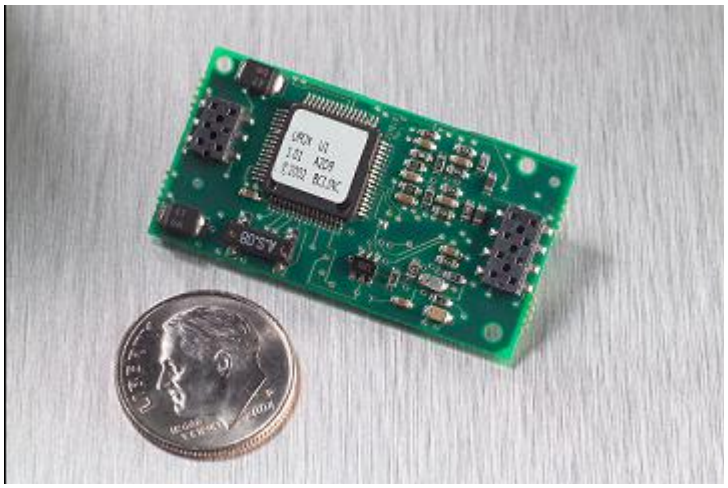
Harvard Üniversitesi, Uygulamalı Bilimler ve Mühendislik Bölümü tarafından geliştirilmiş olan ve yapısında değişik özellikte telsiz algılayıcı düğümler bulunduran bir telsiz algılayıcı ağdır[10]. Projede hedeflenen, hastane dışında ve içinde hastalarla ilgili detaylı bilgilerin toparlanması ve bu bilgiler ışığında daha iyi tedavi edilebilmelerinin sağlanmasıdır.

Proje kapsamında 3 adet farklı özellikte telsiz algılayıcı düğüm geliştirilmiştir. Bu düğümlerden Şekil 3.6'da görülen telsiz "pulse oximeter" düğümü, hastanın kalp atışının ve kanındaki oksijen yoğunluğunun ölçülmesinde kullanılmaktadır.



Şekil 3.6 Pulse oximeter düğümü[10]

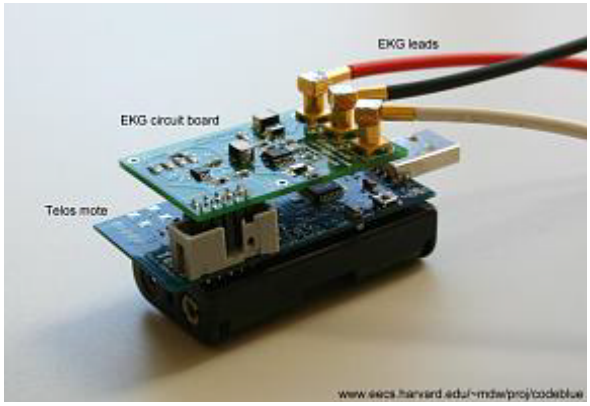
Düğüm, UC Berkeley Üniversitesi tarafından geliştirilmiş olan Mica2 telsiz algılayıcı düğüm üzerine takılan özel algılayıcılar ile oluşturulmuştur. Algılayıcılar bir algılayıcı kart üzerine yerleştirilmiş ve düğüme özel bir genişleme bağlantısı üzerinden bağlanmıştır. Kandaki oksijen miktarının ölçülmesinde kullanılan algılayıcı kulak memesi veya işaret parmağının iki yüzüne yerleştirilen led'ler ve optoelektronik algılayıcılardan oluşmaktadır. Kandaki oksijen miktarı hemoglobin tarafından emilen ışık miktarı ile ölçülmektedir. Şekil 3.7'de düğüm üzerinde kullanılan, Smiths Medical PM şirketi tarafından üretilmiş olan oximeter (oksijen ölçüm) kartı görülmektedir.



Şekil 3.7 Smiths Medical PM'in geliştirdiği oksijen ölçüm kartı[10]

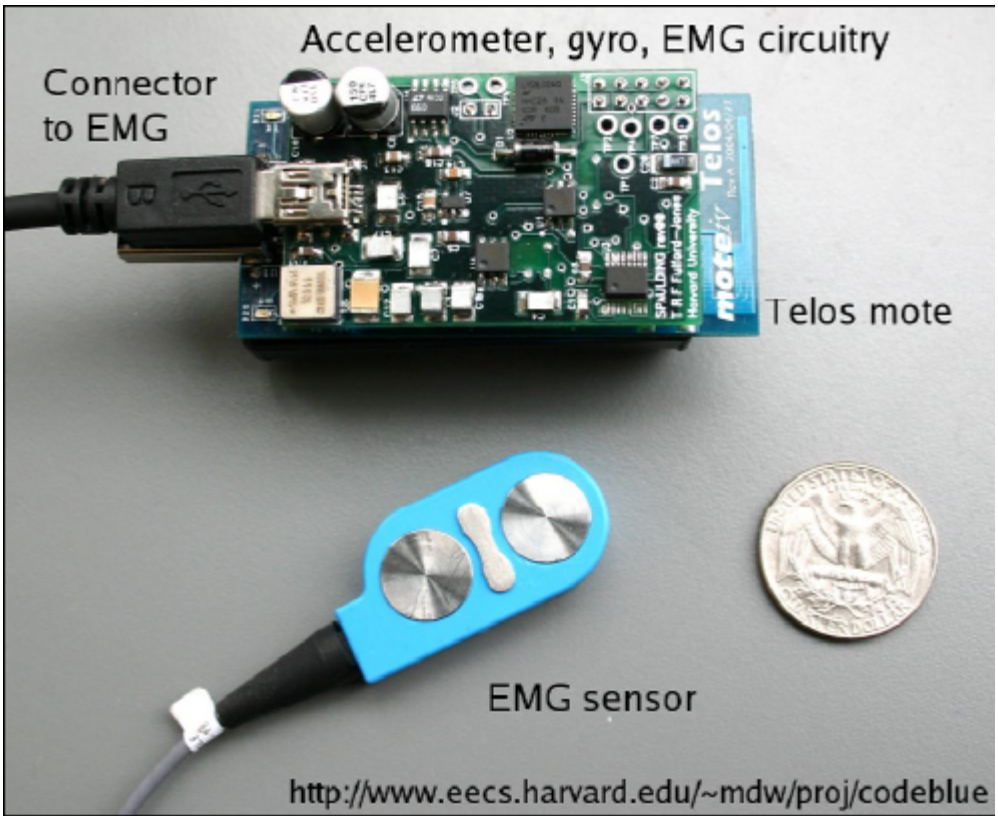
Projede geliştirilen diğer bir düğüm de EKG (Electrocardiograph) ölçümlerinin yapılmasında kullanılmaktadır. Düğüm, Mica2 telsiz algılayıcı düğüme eklenen bir algılayıcı kart ile oluşturulmuştur. Algılayıcı kart üzerinde INA321 operasyonel kuvvetlendirici kullanılarak oluşturulmuş ve MIT Media laboratuarı tarafından geliştirilmiş olan 2 uçlu EKG sistemi bulunmaktadır. Oluşturulan telsiz algılayıcı düğüm Şekil 3.8'de görülmektedir. Düğüm üzerinde bulunan 3 adet uçtan biri üst göğüs diğeri alt göğüs bölgesine yerleştirilir, üçüncü ucun amacı ise sistematik hataları engellemektir. Kullanılan INA321 üretilen sinyali 5 katına büyütür ve oluşabilecek gürültünün çoğunu süzer[10].

Proje kapsamında geliştirilen son düğümün amacı ise felç geçirmiş hastaların iyileşme süreçlerinin ve Parkinson hastalarına uygulanan tedavilerin izlenmesidir. Felç, insan beyninde oluşan bir iç kanama veya beyindeki herhangi bir noktaya yeterli kanın gitmemesi sonucu oluşan bir durumdur. Her iki durumda da beynin belirli bir kısmındaki faaliyet geçici veya kalıcı olarak durur. Parkinson ise genellikle 50'li yaşlarda başlayan beyin faaliyetlerinde düzensizlik sonucu oluşan bir hastalıktır ve en genel belirtisi ellerde oluşan titremedir.



Şekil 3.8 EKG düğümü [10]

Geliştirilen telsiz algılayıcı düğümlerin hastalar üzerine yerleştirilmesi ile hastanın devamlı olarak gözetim altında tutulması ve izlenmesi sağlanmaktadır. Bu şekilde hastanın uygulanan tedaviye (verilen ilacın dozajı ve saati gibi) verdiği tepki en iyi şekilde anlaşılabilir. Kullanılan düğümün içerisinde 3 çeşit algılayıcı bulunmaktadır. Bu algılayıcılar STMicroelectronics tarafından üretilen LIS3L02AQ 3 eksenli ivme ölçer (accelerometer), Analog Devices tarafından üretilen ADXRS300 tek eksenli dönüölçer (gyroscope) ve Motion Lab systems tarafından üretilen MP1A.20.A0DM.60 EMG (ElectroMyoGraph) algılayıcısıdır. Şekil 3.9'da oluşturulmuş olan düğüm görülmektedir.



Şekil 3.9 Felçli hastaların izlenmesi için geliştirilen düğüm[10]

4. KULLANIMDA OLAN TELSİZ ALGILAYICI DÜĞÜMLER

Bu bölümde, 3. bölümde incelenmiş olan telsiz algılayıcı ağlarda kullanılan algılayıcı düğümler ve benzer uygulamalarda kullanılmak üzere tasarlanmış olan ticari ve akademik diğer telsiz algılayıcı düğümler detaylı olarak incelenmiştir.

UC Berkeley tarafından yönetilen telsiz algılayıcı düğüm projesini iki aşamada inceleyebiliriz. Birinci aşama, 1998 ve 2000 yılları arasında WeC, Rene, Rene2 ve Dot kod adlı düğümlerin geliştirildiği dönemdir. İkinci aşama ise 2001 yılından bugüne kadar olan dönemdir. İkinci aşama, düşük güç tüketimine sahip mikro denetleyici ve telsiz iletişim teknolojilerinde yaşanan önemli gelişmeler ışığında çok daha gelişmiş düğümlerlerin oluşturulduğu dönemdir.[11]

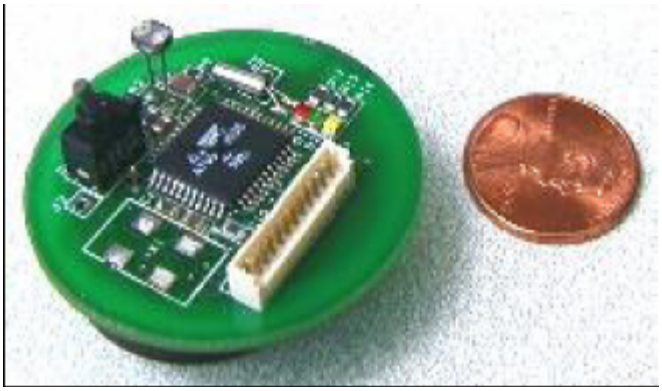
Bu bölümde incelenen diğer araştırma amaçlı telsiz algılayıcı düğümler ise, Avrupa Araştırma projesi olan Eyes kapsamında geliştirilen düğüm, Harvard üniversitesi tarafından CodeBlue projesi kapsamında geliştirilen "Pluto"[10], Hogthrob projesi kapsamında geliştirilen düğüm, California Üniversitesi Center for Embedded Computer Systems bölümü tarafından geliştirilen DuraNode ve Eco'dur.

Ayrıca moteiv şirketi tarafından geliştirilen "tmote Invent"[20] ve "tmote Sky"[20] ile Crossbow şirketi tarafından geliştirilmiş olan MicaZ[21] ticari amaçlı telsiz algılayıcı düğümler incelenmiştir. Bu düğümlerin dışında geliştirilmiş olan başka telsiz algılayıcı düğümler de mevcuttur. Ancak ticari amaçlı olmaları nedeniyle haklarında yeterli bilgi edinilememiştir. National Center for Landscape Fire Analysis tarafından geliştirilen "The Fire Intelligence Module"[37], Tendrill Networks[36] tarafından geliştirilen genel amaçlı telsiz algılayıcı düğüm bunlardan bazılarıdır. Atmel AVR AT90S2313

4.1 Wec, Rene, Rene2, Dot

UC Berkeley üniversitesi tarafından telsiz algılayıcı düğüm tasarlanmasını amaçlayan bir program kapsamında geliştirilen ilk düğüm olan WeC, Şekil 4.1'de görülmektedir. 1998 yılında gerçekleştirilen WeC'in yapısında AT90LS8535 mikrodenetleyicisi ve TR1000 telsiz iletişim birimi kullanılmıştır. Güç kaynağı olarak raf ömrü 10 yıl olan CR2450 lithium pil kullanılmıştır. Kullanılan pilin oldukça küçük olması geliştirilen düğümün boyutlarının da küçük olmasını sağlamıştır. Ancak bu durum düğümün çalışma süresinin kısa olmasına neden olmuştur. Algılayıcı düğümün üzerinde sadece ışık ve sıcaklığın ölçülmesinde kullanılan algılayıcılar bulunmaktadır. Düğüm üzerinde farklı algılayıcıların bağlanabilmesi için

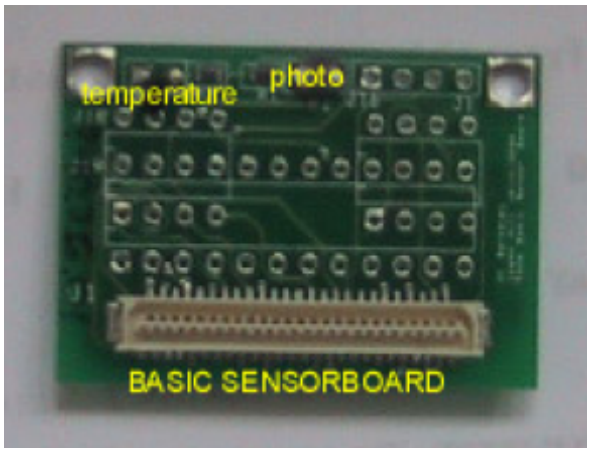
herhangi bir genişleme yuvası bulunmamaktadır.[11]



Şekil 4.1 WeC telsiz algılayıcı düğümü[11]

1999 yılında geliştirilen ikinci düğüm olan Rene Şekil 4.2'de görülmektedir. Rene'nin geliştirilmesinde hedeflenen, farklı tipteki algılayıcıların bağlanmasına imkan veren ve daha uzun çalışma ömrüne sahip bir algılayıcı düğümün oluşturulmasıdır. Bu yüzden pil olarak 2 adet AA boyutunda lithium piller kullanılmıştır. Ancak bu piller düğümün çalışma ömrünü uzatırken boyutlarının da büyümesine neden olmuştur. Düğüm üzerinde farklı algılayıcıların bağlanmasına imkan veren 51 pinli bir genişleme yuvası bulunmaktadır. Şekil 4.3'te bu bağlantı kullanılarak oluşturulmuş bir algılayıcı kart görülmektedir. Bu düğümün hemen ardından 2000 yılında Rene2 düğümü gerçeklenmiştir. Rene ve Rene2 arasındaki temel fark kullanılan mikrodenetleyicidir. Rene2 düğümünde AT90LS8535 yerine AtMega163 mikrodenetleyicisi kullanılmıştır. Bu şekilde işlem kapasitesi arttırılmıştır ve uyanma süresi azaltılmıştır.[11]

Şekil 4.2 Rene telsiz algılayıcı düğümü[11]



Şekil 4.3 Rene için hazırlanmış olan algılayıcı kart[11]

2000 yılında geliştirilen diğer bir düğüm ise Dot isimli düğümdür. Bu düğümün geliştirilme amacı Rene2 ile aynı çalışma performansına sahip ancak daha küçük bir düğümün oluşturulmasıdır. Bu yüzden düğümün boyutlarının büyümesine neden olan AA piller yerine CR2430 yassı piller kullanılmıştır. Üzerinde basınç, sıcaklık ve ivme gibi bilgilerin ölçülmesi için kullanılabilecek algılayıcıların bağlanması için 4 adet bağlantı yuvası bulunmaktadır. Şekil 4.4'te düğümün resmi görülmektedir.

WeC, Rene, Rene2 ve Dot düğümleri günümüzün ihtiyaçlarına artık karşılık verememektedir. Ancak oldukça düşük olan maliyetleri, çok sayıda telsiz algıyacı düğümden oluşan ve rotalama algoritmalarının incelenmesinde kullanılacak test ağlarının oluşturulmasında kullanılmalarını sağlamaktadır.

Şekil 4.4 Dot telsiz algılayıcı düğümü[11]

4.2 Mica, Mica2Dot, Mica2, Telos

UC Berkeley projesinin ikinci kısmında oluşturulan düğümlerden ilki Mica düğümüdür.

Mikro denetleyici birimi olarak AtMega163'e göre güç tüketimi daha az olan AtMega128 kullanılmıştır. İkincil bellek birimi olarak ise daha önceki düğümlerde kullanılan 24LC256'ya göre çok daha fazla kapasiteye sahip AT45DB041B kullanılmıştır.[11]

Mica2 ve Mica2Dot düğümlerinde ise Mica'dan farklı olarak telsiz iletişim birimi olarak CC1000 kullanılmıştır. Mica2Dot' un geliştirilme amacı daha küçük piller kullanılarak küçük boyutlu düğümlerin oluşturulmasıdır.

Proje kapsamında geliştirilen son düğüm olan Telos ise tüm özellikleri ile geliştirilen diğer düğümlerden ayrılmaktadır. Düğümün yapısında, mikrodenetleyici olarak güç tüketimi AtMega128'in yaklaşık onda biri olan MSP430 kullanılmıştır. Telsiz iletişim birimi olarak ise veri aktarım hızı TR1000 ve CC1000'e göre yaklaşık 6 kat fazla olan CC2420 kullanılmıştır.[11]

UC Berkeley tarafından geliştirilen tüm düğümlerin Şekil 4.5'te detaylı olarak karşılaştırılması bulunmaktadır.

Mote Type Year	*WeC* 1998	*René* 1999	*René 2* 2000	*Dot* 2000	*Mica* 2001	*Mica2Dot* 2002	*Mica 2* 2002	*Telos* 2004
								
Microcontroller								
Type	AT90LS8535		ATmega163		ATmega128			TI MSP430
Program memory (KB)	8		16		128			60
RAM (KB)	0.5		1		4			2
Active Power (mW)	15		15		8		33	3
Sleep Power (μW)	45		45		75		75	6
Wakeup Time (μs)	1000		36		180		180	6
Nonvolatile storage								
Chip	24LC256				AT45DB041B			ST M24M01S
Connection type	I^2C				SPI			I^2C
Size (KB)	32				512			128
Communication								
Radio	TR1000				TR1000	CC1000		CC2420
Data rate (kbps)	10				40	38.4		250
Modulation type	OOK				ASK	FSK		O-QPSK
Receive Power (mW)	9				12	29		38
Transmit Power at 0dBm (mW)	36				36	42		35
Power Consumption								
Minimum Operation (V)	2.7		2.7			2.7		1.8
Total Active Power (mW)	24				27	44	89	41
Programming and Sensor Interface								
Expansion	none	51-pin	51-pin	none	51-pin	19-pin	51-pin	10-pin
Communication	IEEE 1284 (programming) and RS232 (requires additional hardware)							USB
Integrated Sensors	no	no	no	yes	no	no	no	yes

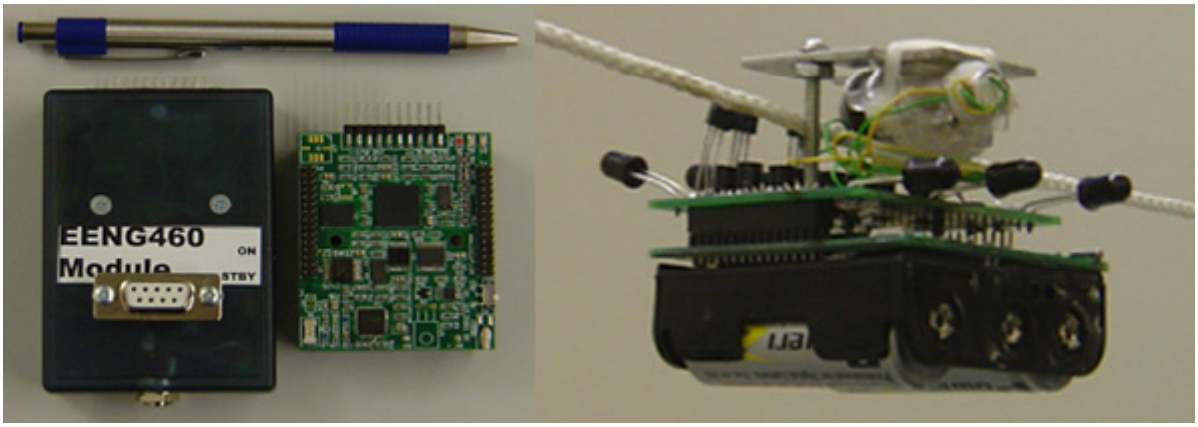
Şekil 4.5 UC Berkeley tarafından geliştirilmiş olan telsiz algılayıcı düğümlerin karşılaştırılması[11]

4.3 XYZ

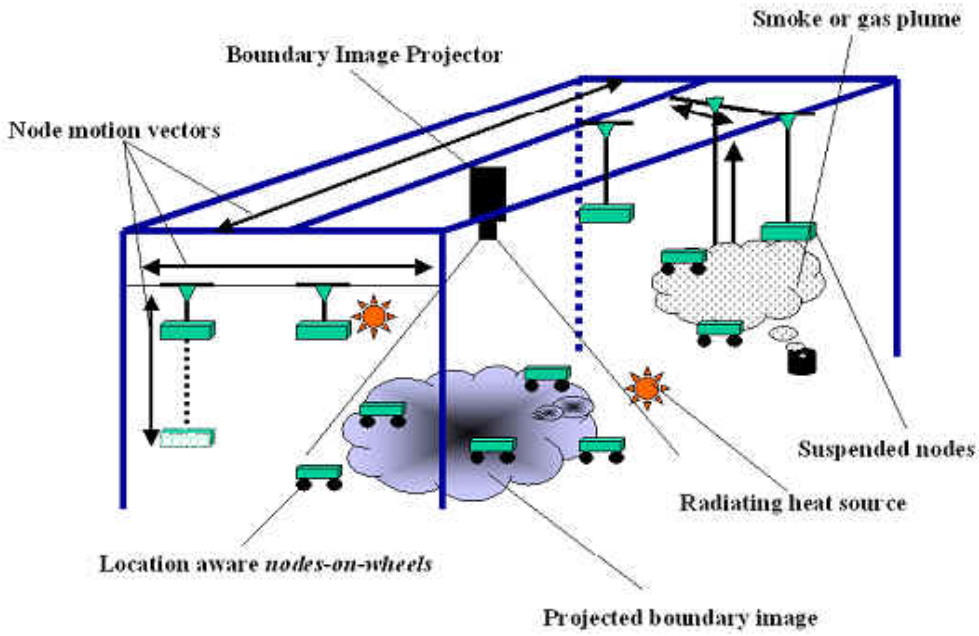
Bu düğüm Yale Üniversitesi, Embedded Networks and Applications Laboratuarı ve Cogen Computer firması tarafından geliştirilmiştir. Deneylerde, eğitim amaçlı projelerde ve endüstriyel uygulamalarda kullanılmak üzere geliştirilmiştir.[16]

Düğümün kullanım amacı ise telsiz bir algılayıcı ağ içerisinde bulunan düğümlerin yerlerinin belirlenmesi (node localization), sınır kestirimi (boundary estimation) ve düğüm ayarlanmasıdır (node calibration).

Düğümün işlemci birimini OKI ARM ML67Q500x serisi bir işlemci oluşturmaktadır. Telsiz iletişim birimi olarak ise CC2420 kullanılmıştır. Düğümler üzerinde sıcaklık, hız ve ışığın ölçülmesinde kullanılan algılayıcılar bulunmaktadır. Geliştirilen sistemin beslenmesi için 3 adet AA boyutunda pil kullanılmaktadır.[16] Bu da düğümün boyutlarının oldukça büyümesine neden olmaktadır. Düğüm Şekil 4.6'da görülmektedir.



Şekil 4.6 XYZ düğümüne ait çeşitli resimler[16]



Şekil 4.7 ENALAB içerisinde oluşturulan telsiz algılayıcı ağın yapısı[16]

ENALAB içerisinde 100 adet XYZ düğümünden oluşan bir test ağı kurulmuştur. Oluşturulan ağ Şekil 4.7'de görülmektedir.

4.4 Mulle

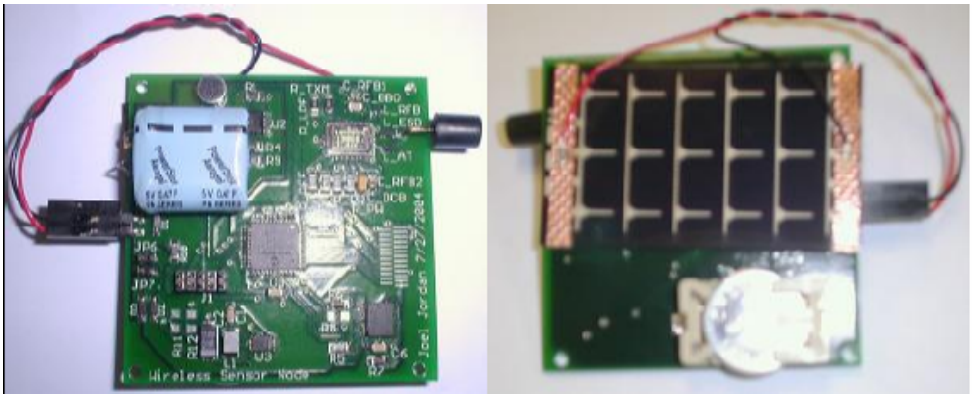
Bu düğüm LULEA Teknik Üniversitesi, Bilgisayar Bilimleri ve Elektrik Mühendisliği bölümü bünyesinde bulunan EISLAB (Embedded Internet System Laboratory) tarafından geliştirilmiştir. Düğümün iki farklı modeli vardır. Birinci model prototip geliştirilmesinde kullanılan ve çok sayıda genel amaçlı giriş çıkış uçlarına sahiptir. İkinci model ise üzerinde ivme, sıcaklık ve ışık algılayıcılarının bulundurur.[17]

Şekil 4.8 Mulle düğümünün boyutlarını gösteren şekil[17]

Geliştirilen düğümü, diğer düğümlerden ayıran özellikleri, iletişim birimi olarak Bluetooth teknolojisinden faydalanması ve tasarlanan kartın 6 katmana sahip olmasıdır.

4.5 UIUC – Sensor Node

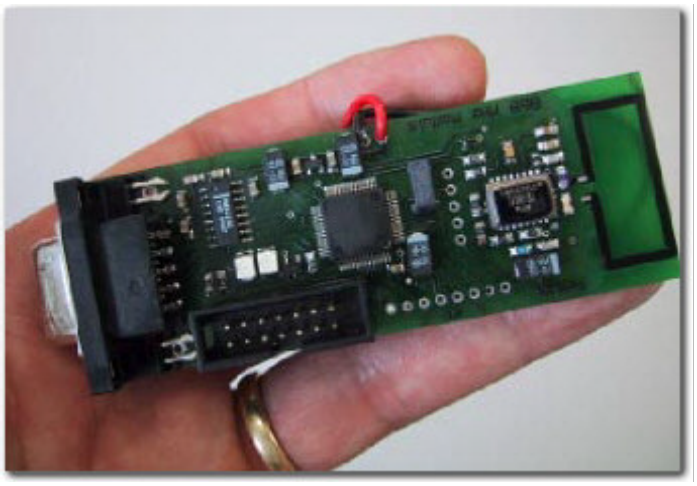
Illinois Üniversitesi tarafından kendi kendine güç üreten telsiz bir algılayıcı tasarlanması amacıyla başlatılan bir proje kapsamında geliştirilmiştir. Mikro denetleyici birimi olarak PIC18F4320'nin kullanıldığı düğümünün kendi enerjisini sağlaması için güneş panelleri kullanılmıştır.[18] Oluşturulan düğümün üstten ve alttan görünüşü Şekil 4.9'dadır.



Şekil 4.9 Illinois Üniversitesi tarafından geliştirilen düğümün üstten ve alttan görünüşü[18]

4.6 Eyes

EYES 3 yıllık bir Avrupa araştırma projesidir. Projede hedeflenen kendi kendine organize olan (self-organizing) ve düşük güç tüketimli telsiz algılayıcı ağ oluşturmaktır. Proje 2002 yılında başlayıp 2005 yılının Şubat ayında bitmiştir. Bu zaman zarfında oluşturulacak ağ içerisinde kullanılacak olan düğümün bir prototipi üretilmiştir. Prototip Şekil 4.10'da görülmektedir.[19]



Şekil 4.10 Eyes projesi kapsamında geliştirilen prototip düğüm[19]

Geliştirilen düğümün mikrodenetleyici birimini MSP430F149 oluşturmaktadır. Telsiz iletişim birimi olarakta TR1001 kullanılmıştır. Ayrıca düğüm üzerinde kullanılacak işletim sistemi ve çevreden toplanan bilgilerin tutulması için ikincil bellek birimi mevcuttur.

4.7 Tmote Invent

Bu düğüm, moteiv firması tarafından geliştirilmiştir. Düğümün mikrodenetleyici birimi olarak MSP430, kablosuz iletişim birimi olarak ise CC2420 kullanılmıştır. Düğümün üzerinde bulunan piller, düğümü USB portu üzerinden tekrar doldurulabilmektedir. Endüstriyel, güvenlik ve bina izleme uygulamaları için geliştirilmiş olan düğüm üzerinde ışık, sıcaklık, ivme ve ses algılayıcıları bulunmaktadır. Düğümün kullanıcı ile etkileşimini sağlamak için düğüm üzerinde çeşitli işlemlerin gerçekleştirilmesinde veya izlenmesinde kullanılan 3 adet ışık, 1 adet hoparlör ve 2 adet düğme bulunmaktadır. Düğüm menzili kapalı alanlarda 50m, açık alanlarda ise 125m'dir.[20] Düğüm Şekil 4.11'de görülmektedir.

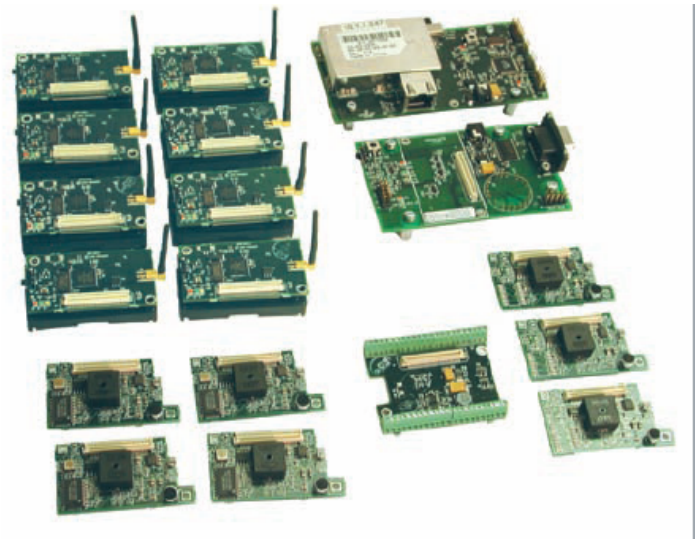
Şekil 4.11 tmote invent telsiz algılayıcı düğüm[20]

Düğüm üzerinde TinyOS işletim sisteminin moteiv firması tarafından değiştirilmiş bir sürümü olan Boomerang işletim sistemi kullanılmaktadır.

4.8 MicaZ

Crossbow firması tarafından geliştirilen düğüm üzerinde TinyOS işletim sisteminini 1.1.7 versiyonu çalışmaktadır. Düğüm üzerinde MSP430 mikrodenetleyici birimi ve CC2420 telsiz iletişim birimi kullanılmıştır. Düğüm üzerinde algılayıcıların bağlanabilmesi için bir bağlantı birimi vardır. Bu bağlantı birimi aracılığıyla Şekil 4.12'de alt kısımda görülen 2 farklı tipteki algılayıcı kart düğüme takılabilmektedir. Bu algılayıcı kartlar üzerinde ivme, manyetizma, ışık, sıcaklık ve akustik algılayıcılar bulunmaktadır. Düğüm Şekil 4.12'de sol üst köşede

görülmektedir.[21]



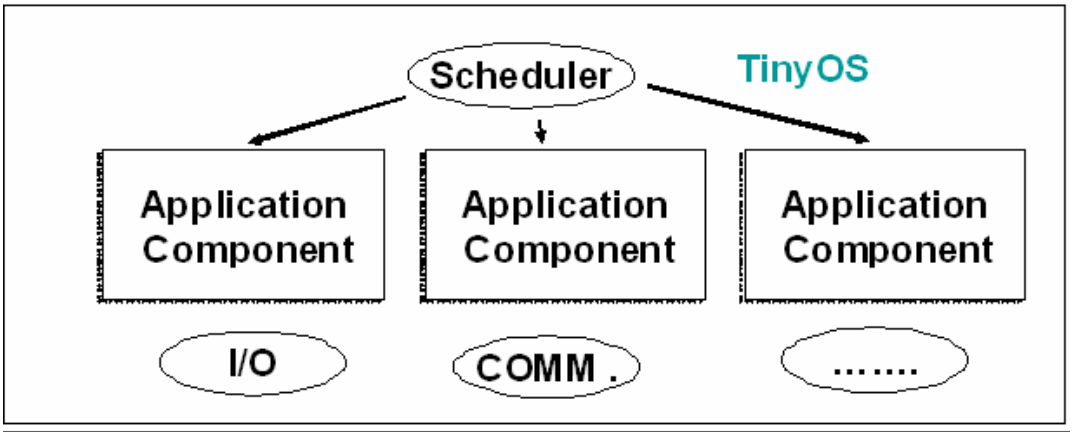
Şekil 4.12 MicaZ düğümleri ve çeşitli algılayıcı kartlar[21]

5. DÜĞÜMLER İÇİN GELİŞTİRİLMİŞ İŞLETİM SİSTEMLERİ

Günümüzde telsiz algılayıcı düğümlerin boyutlarının küçülmesi ve çalışma ömürlerinin yıllar mertebesinde olması istenmektedir. Düğüm boyutlarının küçülmesi, düğümlerin güç ihtiyaçlarının karşılanmasında kullanılan pillerin de küçük boyutlarda olmasını zorunlu kılmaktadır. Bu yüzden düğümlerin kesintisiz çalışma ömürleri birkaç gün ile sınırlıdır. Düğümlerin çalışma zamanlarını uzatabilmek için, çeşitli yazılımsal yöntemler geliştirilmiştir. Bu yazılımsal çözümler arasında telsiz algılayıcı düğümler için geliştirilmiş olan TinyOS[11], MANTIS[23] ve SOS[24] gibi açık kaynaklı, MICROLAN BUCOS ve AVRX[25] gibi ticari işletim sistemleri de bulunmatadır.

5.1 TinyOS

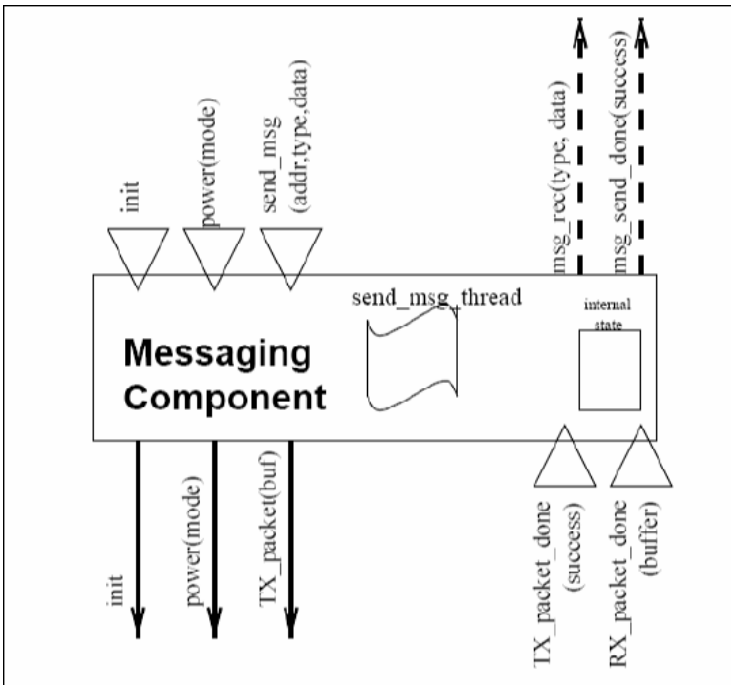
UC Berkeley Üniersitesi tarafından algılayıcı düğümler üzerinde kullanılmak üzere geliştirilmiştir. TinyOS, bileşen tabanlı (component-based) bir mimariye sahiptir. Bu sayede hem hızlı yazılım geliştirilebilmekte hem de algılayıcı düğümlerin kısıtlı bellekleri için uygun kod üretilebilmektedir. TinyOS bünyesinde bulunan bileşen tabanlı kütüphaneler kullanılarak uygulamaya özel çözümler üretilmektedir. Bu kütüphaneler arasında iletişim protokolleri, algılayıcı sürücüleri ve veri toplama araçları (data acquisition) bulunmaktadır. Şekil 5.1'de TinyOS'un bileşen tabanlı mimarisi görülmektedir.



Şekil 5.1 TinyOS işletim sisteminin kaynak yönetimi[22]

TinyOS'un temel hedefi üzerinde çalıştırılacağı telsiz algılayıcı düğümlerin enerji tüketimini en aza indirmektir. Yapısında bir çekirdek (kernel) bulunmamaktadır, uygulamalar donanımsal kaynaklara direkt olarak erişir. Uygulamaların kullanacağı bellek miktarı derleme (compile) işlemi sırasında belirlenir, böylece dinamik bellek tahsisi (dynamic memory allocation) işlemine gerek kalmamaktadır.[22]

Geliştirilmiş olan yapıda herhangi bir anda sadece bir işlem (process) çalışabilmektedir. Bu yüzden oldukça basit bir yöntem olan FIFO (First In First Out) işlem yöntemi kullanılmıştır. TinyOS'un yapısında 2 seviyeli (2 Level) bir zamanlama (scheduling) yöntemi kullanılmıştır. Bu yöntemdeki birinci grubu görevler (tasks), ikinci grubu ise olaylar (events) oluşturmaktadır. Görevler hesaplamaların yapılmasında, olaylar ise veri akışının kontrol edilmesinde kullanılmaktadır. Geliştirilen yapıda olaylar, başka olayları veya görevleri oluşturabilirler. Ancak görevler, başka görevleri veya olayları oluşturamazlar. Olaylar donanımsal kesmeler (interrupts) tarafından oluşturulan düşük seviyeli olaylar ve yazılımlar tarafından oluşturulan yüksek seviyeli olaylar olarak ikiye ayrılmıştır. Olaylar fonksiyon çağırılması (function calls) ile düşük seviyeden yüksek seviyeye doğru yayılabilirler. TinyOS'un yapısında yüksek seviyedeki bileşenler daha düşük seviyedeki bileşenlere komutlar (commands) araçılığıyla istekte bulunur. [22]



Şekil 5.2 İletişim bileşeninin grafiksel gösterimi[22]

Şekil 5.2'de iletişim bileşeninin (messaging component) grafiksel gösterimi bulunmaktadır. TinyOS'un yapısını oluşturan yazılımsal bileşenler genelikle, bir çerçeve (frame), olay işleyici (event handler), komutlar ve görevlerden oluşmaktadır. Şekil 5.2'de gösterilen iletişim bileşeni başlama durumuna getirme, güç yönetimi ve paket gönderilmesi için daha düşük seviyedeki bileşenlere gönderilecek olan komutlara (aşağı doğru ok işaretleri) ve üst seviyeden gelen istekleri işleyecek komutlara (aşağı doğru üçgenler) sahiptir. İletişim bileşeni, iletiminin tamamlanması ve paket alımının tamamlanmasıyla ilgili olayların gerçekleştirilmesinden sorumludur. Bu olayların gerçeklendiğinde düşük seviyelerden gelen olay sinyalleri Şekil 5.2'de üçgenlerle gösterilmiştir. Gerçekleşen olayların üst seviyeye bildirilmesi için yukarı ok işretleriyle gösterilen olay sinyalleri kullanılmaktadır. Şekil 5.3'te yukarıda incelenmiş olan iletişim bileşeninin tanımı görülmektedir.[22]

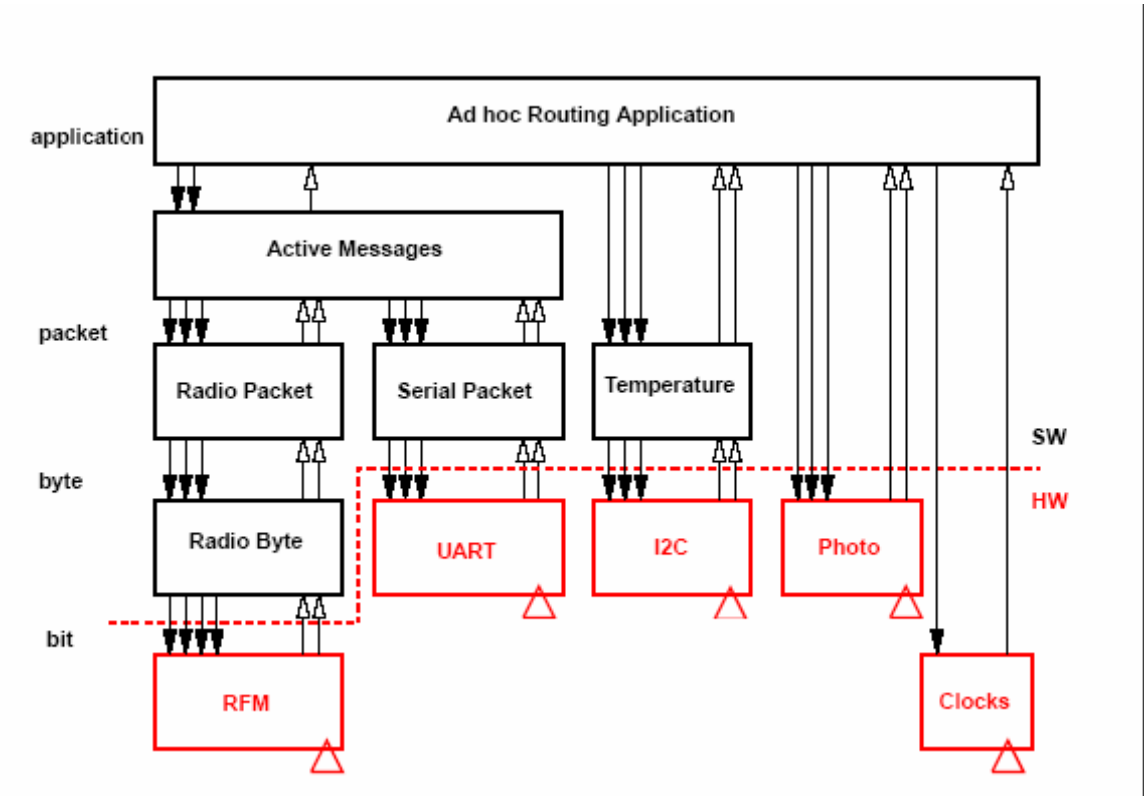
```
ACCEPTS{
        char AM_SEND_MSG(char addr,char type, char* data);
        char AM_POWER(char mode);
        char AM_INIT();
};


SIGNALS{
        char AM_MSG_SEND_DONE(char success);
        char AM_MSG_REC(char* data);
}
HANDLES{
        char AM_TX_PACKET_DONE(char success);
        char AM_RX_PACKET_DONE(char* packet);
};

USES{
        char AM_SUB_TX_PACKET(char* data);
        void AM_SUB_POWER(char mode);
        char AM_SUB_INIT();
};
```

Şekil 5.3 İletişim bileşeninin yazılımsal tanımlaması[22]

TinyOS'un yapısında bulunan bileşenler 3 ayrı grupta incelenebilir. Bu bileşen grupları; donanım soyutlaması (hardware abstraction), soyut donanım (synthetic hardware) ve üst seviye yazılımsal (high level software) bileşenlerdir. Donanım soyutlama bileşenleri, fiziksel donanımların TinyOS yapısını oluşturan bileşen modeline eklenmelerini sağlar. Soyut donanım bileşenleri ise gelişmiş donanımların simüle edilmesini sağlar. Bu çeşit bir bileşene en iyi örnek Radio Byte bileşenidir. Bu bileşen telsiz iletişim birimine verinin byte olarak aktarılması ve bu birimden bilginin byte olarak alınmasından sorumludur. Üst seviye yazılımsal bileşenler ise kontrol, rotalama ve veri aktarımı gibi işlemlerden sorumludur. Şekil 5.4'te TinyOS'un çeşitli bileşenleri kullanılarak oluşturulmuş örnek bir uygulamanın grafiksel gösterimi verilmiştir. [22]



Şekil 5.4 TinyOS bileşenleri ile oluşturulan örnek bir uygulama[22]

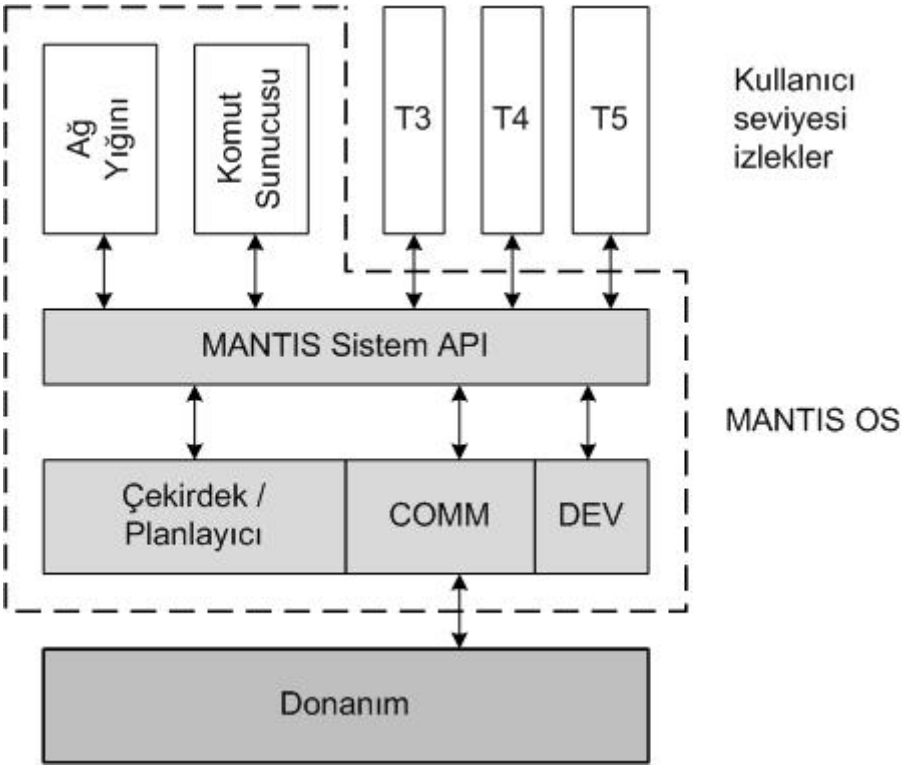
TinyOS işletim sistemi birçok değişik donanım üzerinde çalışabilmektedir ve birçok projede kullanılmaktadır. Bu donanımlardan bazıları yine UC Berkeley Üniversitesi tarafından geliştirilmiş olan WeC, Rene, Rene2, Dot, Mica, Mica2Dor, Mica2 ve Telos'tur. TinyOs'un kullanıldığı projelerden bazıları, Calamari[31], CotsBots[32], Great Duck Island[2], Firebug[33], TinyDB[34], TinyGALS[35]'tır.

5.2 MANTIS

MANTIS, telsiz algılayıcı düğümler için öoklu görevlendirmeli bir işletim sistemi oluşturulması amacıyla Colarado Üniversitesi Bilgisayar Bilimleri Bölümü tarafından geliştirilmiştir. Şekil 5.5'de işletim sisteminin mimarisi görülmektedir. MANTIS projesinde hedeflenen, geliştirilen uygulamaların Palm, PC gibi birçok farklı platformda çalıştırılabilmesine imkan veren bir işletim sisteminin oluşturulmasıdır. Bu yüzden taşınabilir bir programlama dili olan C kullanılarak yazılmıştır. (Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)

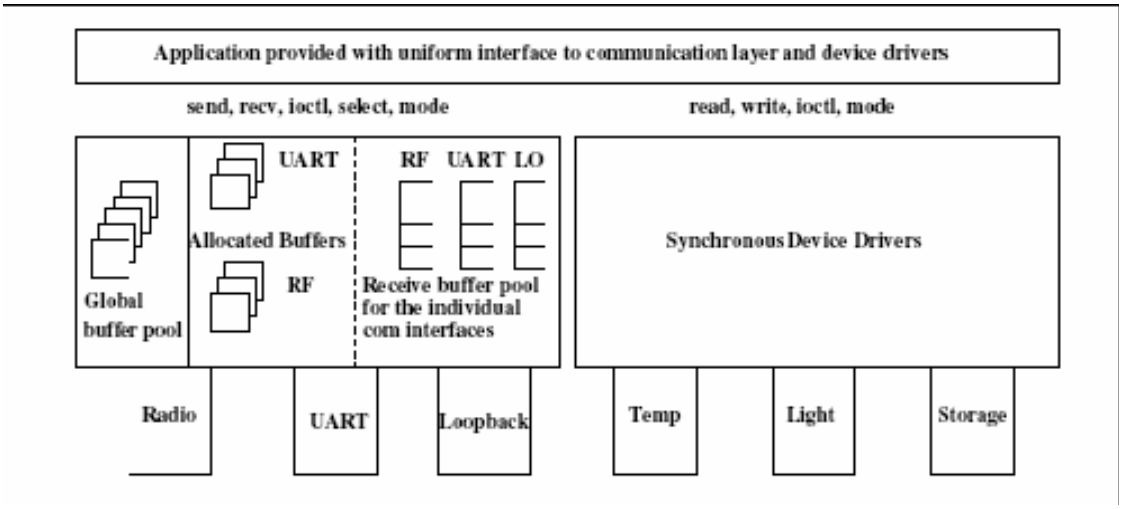
İşletim sisteminin çekirdeği içerisinde Unix tabanlı işletim sistemlerinde kullanılan zamanlayıcılara (scheduler) benzer bir zamanlayıcı kullanılmıştır. Zamanlama yöntemi olarak

öncelik seviyelerine sahip round robin yöntemi kullanılmaktadır. Threadlerin eş zamanlılığının sağlanması için ikili ve sayan semaphore'lar kullanılmaktadır. Zamanlayıcı, threadler arasındaki geçişleri donanımsal olarak oluşturulan bir zamanlayıcı kesmesi ile gerçekleştirir. Bu zaman aralığının normal değeri 10 ms'dir. Ancak bu değer yazılımsal olarak değiştirilebilir. Bu geçişler semaphore işlemleri veya sistem çağrıları ile de gerçekleştirilebilir. Oluşan kesmeler arasında sadece zamanlayıcı kesmesi çekirdek tarafından değerlendirilir. Diğer tüm kesmeler ilgili aygıt sürücülerine yönlendirilir. (Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)



Şekil 5.5 MANTIS işletim sisteminin blok diagramı(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)

Şekil 5.6'da solda radyo, seri ve benzeri birimlerle eş zamansız olarak iletişi sağlayan yapı, sağda ise donanım sürücülerine eş zamanlı olarak erişimi sağlayan yapı görülmektedir. Uygulamanın bu iki yapıya erişimi, bir katman kullanılarak tekdüze bir hale getirilmiştir.



Şekil 5.6 MANTIS içerisindeki iletişim ve sürücü katmanları(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)

Düğüm üzerinde bulunan aygıtlara erişim için POSIX (Portable Operating System Interface) benzeri bir yapı oluşturulmuş ve dev_read(), dev_write(), dev_mode() ve dev_ioctl() sistem çağrıları gerçeklenmiştir. dev_mode() sistem çağrısı güç yönetimi için kullanılır. Düğüm üzerindeki aygıtların açık, kapalı ve pasif durumlarında olabilir. Bu aygıtların durumlarında değişiklik dev_mode() sistem çağrısı yapılarak gerçekleştirilir. dev_ioctl() sistem çağrısı ise aygıtlara özel parametrelerin değiştirilmesinde kullanılır. Örneğin bu sistem çağrısı kullanılarak düğüm üzerinde bulunan bir EPROM'un hangi adresine erişileceği belirlenir. dev_read() ve dev_write() sistem çağrıları düğüm üzerindeki aygıtlardan veri okunması veya bu aygıtlara veri aktarılması için kullanılır. (Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)

Şekil 5.7'de MANTIS işletim sisteminin yapısında bulunan programlama arayüzleri (Application Programming Interface - API) kullanılarak geliştirilmiş olan algıla ve ilet (sense and forward) yapısında bir uygulamaya ait kaynak kod görülmektedir.

```
#include <inttypes.h>
#include "led.h"
#include "dev.h"
#include "com.h"

void start(void)
{
    uint16_t delay;
    uint8_t value;
    comBuf send_pkt;

    value = dev_get(DEV_MICA2_LIGHT);
    mos_led_toggle(0);

    send_pkt.data[0] = value;
    send_pkt.size=1;
    com_send(IFACE_RADIO, &send_pkt);
}
```

Şekil 5.7 MANTIS işletim sistemi için yazılmış örnek bir uygulama(Bhatti, Carlson, Dai, Deng, Rose, Sheth, Shucker, Gruenwald, Torgerson, Han, 2005)

MANTIS işletim sistemi, birçok telsiz algılayıcı düğüm üzerinde çalışabilmektedir. Bu düğümlerden bazıları UC Berkeley tarafından geliştirilmiş olan Mica, Mica2Dor, Mica2 ve Telos'tur.

5.3 SOS

SOS işletim sistemi, UCLA Üniversitesi bünyesinde bulunan NESL (Networked and Embedded Systems Labrotory) tarafından geliştirilmiştir. Geliştirilmesindeki temel amaç telsiz algılayıcı düğümlerin çalışma yapılarının dinamik olarak değiştirilmesidir. Bu yüzden işletim sisteminin çekirdeği, dinamik bellek tahsisine imkan sağlayan modüler bir yapı üzerine kurulmuştur.[24]

SOS işletim sistemi, bir telsiz algılayıcı ağı kurulduktan sonra düğümlerinin farklı şekillerde ayarlanmalarına imkan sağlamaktadır. Bu şekilde her düğüm için ayrı yazılımlar geliştirilmek yerine tek bir yazılım geliştirilip farklı düğümler üzerinde sadece gerekli modüller çalıştırılabilir.[24]

SOS işletim sisteminin çekirdeği dinamik bellek tahsisinin yanısıra çöp toplama (garbage collection) ve öncelikli zamanlayıcı (priority scheduling) özelliklerini de sağlamaktadır. SOS şu an için sadece Crossbow tarafından geliştirilen MicaZ ve Yale üniversitesi tarafından geliştirilen XYZ düğümleri üzerinde çalışmaktadır.

5.4 Ticari İşletim Sistemleri

Telsiz algılayıcı düğümler üzerinde kullanılmak üzere geliştirilmiş olan çeşitli ticari işletim sistemleri bulunmaktadır. Bunlardan biri AVR mikrodenetleyicileri üzerinde çalışacak şekilde Larry Barello tarafından geliştirilmiş olan AVRX'tir. AVRX gerçek zamanlı, çok görevli (multitasking) bir çekirdeğe sahiptir. Çekirdeği assembly programlama dili ile yazılmıştır. Çekirdeğinin yapısında 16 farklı öncelik seviyesinin yer aldığı bir zamanlayıcı bulunmaktadır. Görevler arasında bilgi aktarımı için mesaj kuyrukları (message queue) bulunmaktadır. Şekil 5.8'de AVRX yapısında kullanılmak üzere yazılmış olan bir göreve ait kod görülmektedir. AVRX yapısında kullanılacak olan görevler genellikle bir başlatma işlemi (initialization) ve ardından sonsuz bir döngü içinde yapılan işlemlerden oluşur.[25]

```
Mutex Timeout;

AVRX_TASKDEF(myTask, 10, 3)
{
    TimerControlBlock MyTimer;

    while (1)
    {
        AvrXDelay(&MyTimer, 10); // 10ms delay
        AvrXSetSemaphore(&Timeout);
    }
}
```

Şekil 5.8 AVRX için geliştirilen basit bir görev[25]

AVRX işletim sisteminin şu an kullanımda olan iki ayrı sürümü bulunmaktadır. Bunlardan AVRX 2.3 sadece assembly programlama dili ile yazılım geliştirilmesine imkan vermektedir. AVRX işletim sisteminin 2.6 versiyonu ise C kodu ile yazılım geliştrilmesine imkan sağlamaktadır. Şekil 5.9'da ise AVRX 2.6 üzerinde geliştirilen bir uygulamanın ana kodu görülmektedir.[25]

```
void main(void)                 // Main runs under the AvrX Stack
{
    outp((1<<SE) , MCUCR);      // Enable "Sleep" instruction for idle
loop

    outp(TCNT0_INIT, TCNT0);    // TCNT0_INIT defined in "hardware.h"
    outp(TMC8_CK256 , TCCR0);   // Set up Timer0 for CLK/256 rate
    outp((1<<TOIE0), TIMSK);    // Enable0 Timer overflow interrupt

    outp(-1, LED-1);            // Make PORTB output and
    outp(-1, LED);              // drive high (LEDs off)

    AvrXRunTask(TCB(task1));
    AvrXRunTask(TCB(task2));
    AvrXRunTask(TCB(Monitor));

    InitSerialIO(UBRR_INIT);     // Initialize USART baud rate
generator

    Epilog();                    // Switch from AvrX Stack to first
task
}
```

Şekil 5.9 AVRX 2.6 için yazılmış olan örnek uygulama [25]

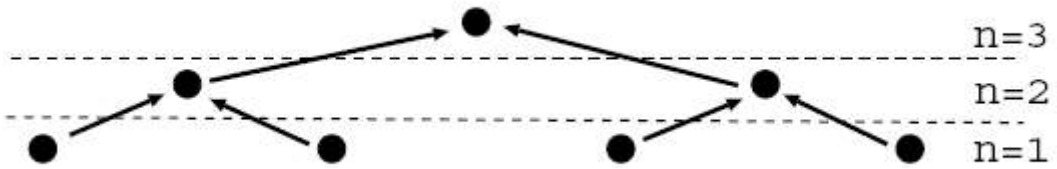
5.5 Olaya Dayalı ve Çoklu Görevlendirmeli İşletim Sistemleri Karşılaştırması

Günümüzde telsiz algılayıcı ağlar üzerinde iki farklı tipte işletim sistemi çalıştırılmaktadır: olaya dayalı (event driven)ve çok izlekli (multi-threaded). Olaya dayalı sistemlerde işletim sisteminin yapması gereken her işlem bir olay tarafından tetiklenir (örn. sayaç bilgisi veya algılayıcılardan bilgi geldiğini bildiren donanım kesmesi). Olaylarla ilgili görevler işletim sistemi boşa çıkıp enerji verimli duruma geçinceye kadar sıralı olarak işlenir. Olaylar sıralı işlendiği için görevler arasında pahalı ortam geçişleri yapılmasına gerek yoktur. Bu işletim sistemlerine örnek olarak TinyOS gösterilebilir. İkinci tip işletim sistemleri ise çok izlekli işletim sistemleridir. İşletim sistemi değişik görevler arasındaki çalışma zamanını çoklar. Bir görevden diğer göreve geçiş yaparken çalışılan ortam kaydedilmeli ve geçiş yapılacak ortam yüklenmelidir. Bu işlem kısıtlı kaynaklarla çalışan telsiz algılayıcı düğümlerin kaynaklarını fazlaca kullanır. Bu tip işletim sistemlerine örnek olarak da MANTIS gösterilebilir. (Duffy, Roedig, Herbert, Sreenan, 2007)

Genel olarak olaya dayalı işletim sistemlerinin az kaynağa ihtiyaç duymasından dolayı kaynak sınırlaması olan telsiz algılayıcı düğümlerde kullanılmasının daha uygun olduğu, çok izlekli işletim sistemlerinin ise zaman sınırlamalı işlemleri yürütmede daha başarılı olduğu farzedilir.

Bu konuyu tam olarak açığa kavuşturmak için DSYS25 [Barroso, Benson, Murphy, Roedig, Sreenan, Barton, Bellis, O'Flynn, Delaney, 2004] algılayıcı platformunda olaya dayalı TinyOS ile çok izlekli MANTIS işletim sistemleri bellek gereksinimleri, enerji harcamaları ve olay işleme yetenekleri açısından test edilmiştir. (Duffy, Roedig, Herbert, Sreenan, 2007)

Şekil 5.10'de görüldüğü üzere ağ içerisinde ikili ağaç topolojisi kullanılmıştır. Ağaçtaki n'in yerine göre algılayıcı düğüm farklı sayıda paket işleyebilir.



Şekil 5.10 İkili Ağaç(Duffy, Roedig, Herbert, Sreenan, 2007)

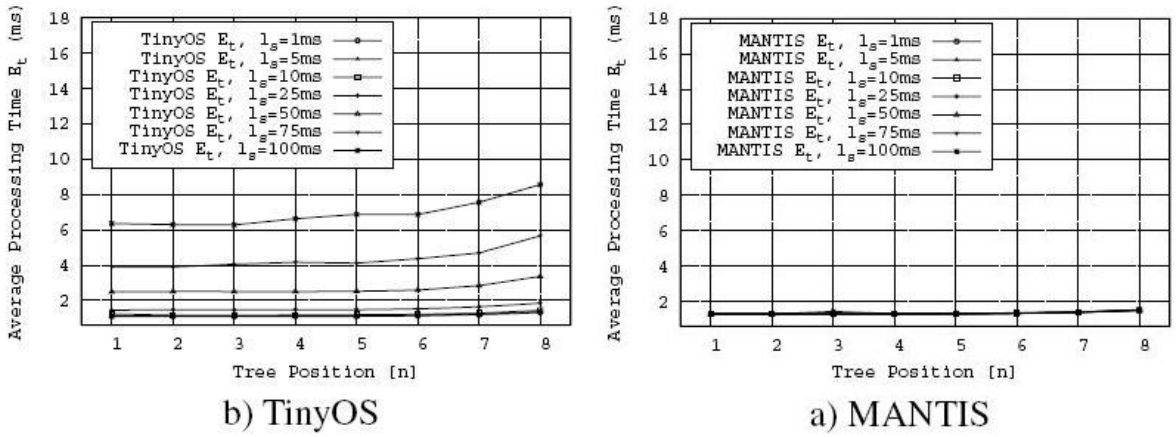
Bir işletim sisteminin bellekte kapladığı alan mümkün olduğu kadar az olmalıdır. Çalıştırılan bir uygulamanın karmaşıklığı ne kadar artarsa, bu uygulamanın çalıştırılabilmesi için gereken en az bellek/işlemci çipi gereksinimi de o kadar artacaktır. Çalıştırılan test programda işletim sistemlerinin bellek kullanımını ölçmek için GNU ikili proje yardımcı programı avr-size kullanılmıştır. Tablo 5.1 MANTIS işletim sisteminin TinyOS'a nazaran %30 daha fazla bellek alanı kapladığını, fakat RAM gereksinimlerinin iki işletim sisteminde neredeyse eşit olduğunu göstermektedir. (Duffy, Roedig, Herbert, Sreenan, 2007)

Çizelge 5.1 Bellek Kullanımları(Duffy, Roedig, Herbert, Sreenan, 2007)

İşletim Sistemi	Program Büyüklüğü (KB)	RAM Gereksinimi (B)
TinyOS	9	283
MANTIS	13.1	287

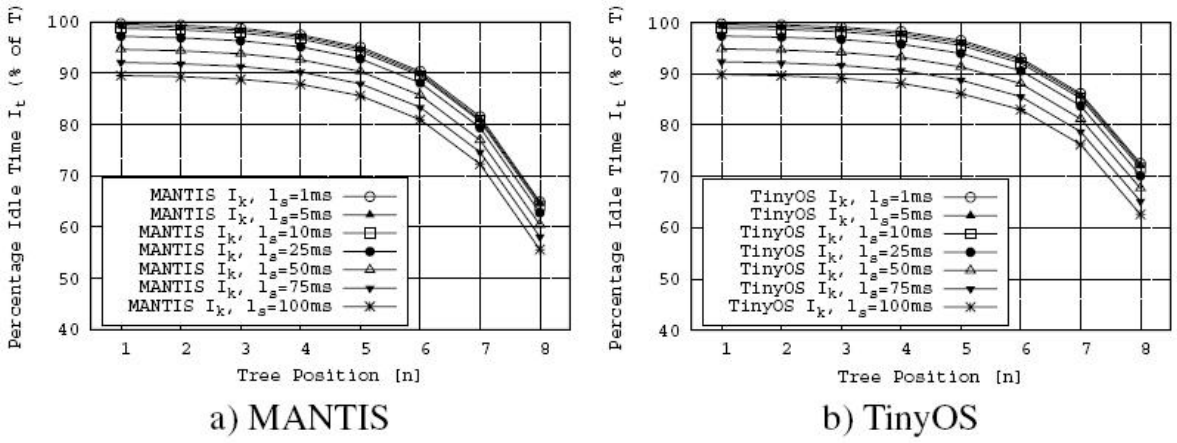
Paket iletiminde zaman sınırının aşılmaması için düğümlerdeki paket işleme görevlerinin önceliğe sahip olması gerektiği farzedilir. MANTIS işletim sisteminde paket işleme görevi algılama görevinden daha önceliğe sahiptir. TinyOS işletim sisteminde ise herhangi bir görev için öncelik tanımlanmamıştır. İşletim sisteminin işlem yapma performansını belirleyebilmek için paket işleme görevinin ortalama görev bitirme zamanı E_t ölçülmüştür. Test işlemi boyunca J adet paket işleme süresi e_j kaydedilmiştir. Ortalama görev bitirme süresi testin sonunda $E_t = \sum e_j / J$ işlemi ile hesaplanmıştır. Ağacın her n seviyesi için paket işleme

görevleri J = 25000 oluncaya kadar kaydedilmiştir. MANTIS işletim sistemi kullanıldığı zaman ortalama paket işleme zamanının algılama görevi bitirme zamanından bağımsız olduğu tespit edilmiştir. Ayrıca E_t, düğümün ağacın hangi n seviyesinde olduğundan da bağımsızdır. Ağır yük altında ortalama işlem süresi, paket iletiminin sıralanması gerektiğinden biraz artmıştır. TinyOS kullanıldığında ise, E_t zamanı algılama görevinin l_s süresine bağımlıdır. Ayrıca ağır yük altında paket işleme görevlerinin sıralanma etkisi de ortalama işlem zamanını artırmıştır. (Duffy, Roedig, Herbert, Sreenan, 2007)



Şekil 5.11 İkili Ağaç(Duffy, Roedig, Herbert, Sreenan, 2007)

Algılayıcı ağların üzerinde çalışan işletim sistemlerinin az enerjiyle çalışması önemli bir faktördür. Bu yüzden, düşük güç gerektiren operasyonların zamanlanabilmesi için işletim sisteminin boş kalan zamanları (idle time) araştırılmıştır. T test işlemi süresince, K defa i_k boş süresi kaydedilmiş ve işletim sistemin boş süre yüzdesi $I_t = (\sum i_k / T) * 100$ formülüyle hesaplanmıştır. Her iki işletim sisteminde de ağaçtaki n seviyesi arttıkça sistemin boş durumda geçirdiği zaman eksponansiyel olarak azalmıştır. Paket görev sayısı arttıkça bu durumun ortaya çıkması beklenen bir durumdur. Fakat MANTIS işletim sisteminde sürekli ortam geçişi yapılması boş süresini daha da düşürmektedir. Şekil 5.12'den de anlaşılabileceği gibi TinyOS, MANTIS işletim sistemine nazaran özellikle ağır yük altında enerjiyi daha yüksek verimde kullanmaktadır. Bu da daha TinyOS'un daha az güç tüketimi yaptığını gösterir. (Duffy, Roedig, Herbert, Sreenan, 2007)



Şekil 5.12 Boşta Kalma Zamanı Yüzdesi I_t (Duffy, Roedig, Herbert, Sreenan, 2007)

Özetlemek gerekirse, MANTIS işletim sistemi TinyOS'a göre %30 daha fazla alana ihtiyaç duymasına rağmen daha öngörülebilir bir sistemdir. Özellikle, paket iletim görev zamanı fazla değişmemekte ve algılama görevi gibi diğer görevlerden bağımsızdır. Bu yüzden MANTIS belirleyici ve zamana bağlı uygulamalarda daha tercih edilebilirdir. Fakat test uygulamasından da görülebileceği üzere TinyOS kadar enerji verimli değildir. Bu sebepten TinyOS enerji tüketiminin birinci önceliğe sahip olduğu durumlarda tercih edilmektedir. (Duffy, Roedig, Herbert, Sreenan, 2007)

6. İŞLETİM SİSTEMİNİN ÇALIŞACAĞI DÜĞÜMÜN DONANIM ELEMANLARI

İşletim sistemin üzerinde çalışacağı algılayıcı düğüm (Tayşi, Z. Cihan, 2006) tarafından yüksek lisans tezi kapsamında geliştirilmiştir ve ismi vf-1a'dır. Düğüm; mikrodenetleyici birimi, telsiz iletişim birimi, ikincil bellek birimi, güç yönetim birimi ve algılayıcılar olmak üzere 5 birimden oluşmuştur.

6.1 Mikrodenetleyici Birimi

vf-1a algılayıcı düğümü üzerinde bulunan mikrodenetleyici birimi Texas firması tarafından geliştirilmiş olan MSP430F149'dur. (Tayşi, 2006)

Güç tüketimi olarak incelediğimiz zaman MSP430F149, 5 adet farklı çalışma seviyesine sahiptir. Bu çalışma seviyelerinin tamamında mikrodenetleyicinin çekirdeği (core) kapalıdır. Bu çalışma seviyeleri kullanılarak mikrodenetleyicinin farklı çevre birimlerinin ve osilatorlerin çalışır veya kapalı konumda olması sağlanabilir. Mikrodenetleyici bu çalışma seviyelerinin hepsinden bir kesme ile normal çalışma seviyesine döner, kesme işlemi bittiğinde yine aynı çalışma seviyesine döner.[12]

MSP430F149 mikrodenetleyicisi en fazla 8MHz hızında çalışmaktadır. Kapalı konumdayken 0.5µA akım çeken bu mikrodenetleyici, işlem yapmadığı boş konumda 55µA, 32KHz hızında çalışırken 19.2µA, 1MHz hızında çalışırken 420µA, 8MHz hızında çalışırken ise 1.9mA akım çekmektedir. [12]

MSP430F149 saat ölçeklenmesi işlemi için çalışma seviyeleri arasında geçiş süresinin (wake-up time) çok düşük olmasını sağlayan FLL yöntemini kullanmaktadır. Bu mikrodenetleyicinin çalışma seviyeleri arasındaki geçiş süresi 6 µs'den daha azdır.[12]

MSP430F149 mikrodenetleyicisi, RISC mimarisinde olup 8 bit kelime uzunluğuna sahip mikrodenetleyicilere göre daha yüksek performansa sahip 16 bit kelime uzunluğuna sahiptir. 51 komut içeren mikrodenetleyicide 16 adet saklayıcı (register) bulunmaktadır.[12]

MSP430F149 mikrodenetleyicisi, 12 bit Analog Sayısal Çeviriciye ve 2 adet SPI, 1 adet UART, 2 adet USART, 1 adet Donanım Çoğaltıcı ve 48 adet genel G/Ç çevre birimine sahiptir.[12]

6.2 Telsiz İletişim Birimi

vf-1a algılayıcı düğümü üzerinde Chipcon firması tarafından üretilen CC2420 telsiz iletişim

birimi bulunmaktadır.(Tayşi, 2006)

CC2420 telsiz iletişim biriminin çalışma frekansı 2483.5MHz olup veri aktarım hızı 250kbps'dir. CC2420'nin çıkış gücü 0dBm ile -25dBm arasındaki değiştirilebilmekte böylece güç tüketimi 8.5 mA'e kadar düşürülebilmektedir. Alıcı hassasiyeti -95dBm olan CC2420 boş durumdayken 426µA, iletim yaparken 17.4mA, alım yaparken de 19.4mA akım çekmektedir.[15]

6.3 İkincil Bellek Birimi

vf-1a algılayıcı düğümü üzerinde ST Microelektronics firması tarafından geliştirilen M25P80 ikincil bellek birimi kullanılmıştır. (Tayşi, 2006)

SPI bağlantı tipine sahip M25P80'in belleği 8192Kbit'dir. Bu ikincil belleğin yazma süresi 1.5ms gibi oldukça hızlı bir değerdir. 25000Khz veri aktarım hızına sahip olan M23P80, pasif durumdayken 1µA, okuma anında 4000µA, yazma anında ise 15mA akım çekmektedir.[26]

6.4 Güç Yönetim Birimi

vf-1a algılayıcı düğümü üzerindeki güç birimi oldukça basit bir yapıya sahiptir. Güç yönetimi için özel bir donanım kullanılmamıştır. Mikrodenetleyici oldukça basit bir yöntemle bu işlemi gerçekleştirmektedir. Mikrodenetleyici sadece ihtiyaç duyulan birimlerin açık olmasını, diğer birimlerin ise kapalı kalmasını sağlamaktadır. (Tayşi, 2006)

Sistemin ihtiyaç duyduğu güç için 3.0V gerilim üreten Lithium-ion yassı (button) pil kullanılmıştır. Ayrıca testler sırasında düğümün kesintisiz çalışmasını sağlamak amacıyla, düğümün dışardan bir güç üretecine bağlanmasını sağlayan bir bağlantı birimi de mevcuttur. (Tayşi, 2006)

6.5 Algılayıcılar

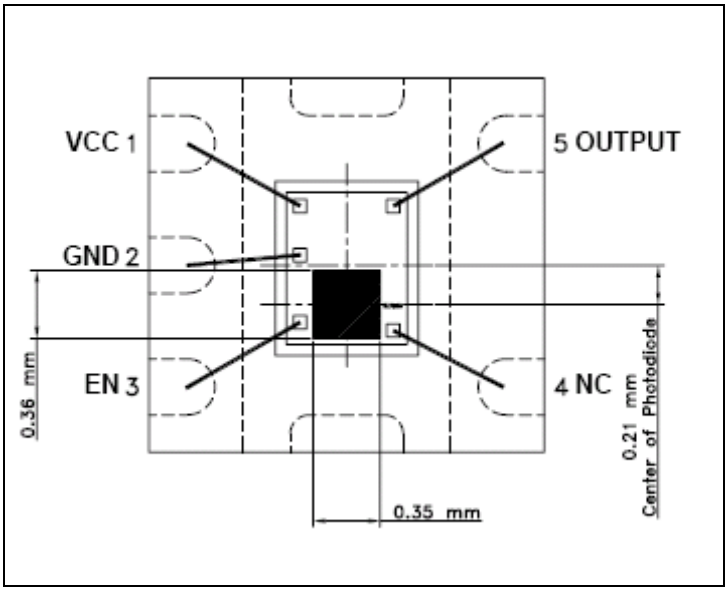
vf-1a algılayıcı düğüm üzerinde bulunan algılayıcılar bugüne kadar oluşturulmuş olan telsiz algılayıcı ağlarda kullanılan düğümler üzerinde bulunan algılayıcılar sınıflandırılarak seçilmiştir. Bugüne kadar oluşturulan telsiz algılayıcı ağlar çevreden genellikle sıcaklık, ışık, nem, basınç, ivme ve manyetik alan bilgileri toplamaktadırlar. Geliştirilmiş olan düğümün çok amaçlı olması için tüm bu bilgileri çevreden toplayabilecek algılayıcılar kullanılmıştır. Çizelge 6'de, yukarıda sayılan fiziksel büyüklüklerden herbirinin çevreden toplanması için kullanılmış algılayıcıların listesi verilmiştir. (Tayşi, 2006)

Çizelge 6.1 Kullanılan algılayıcıların listesi (Tayşi, 2006)

Fiziksel Büyüklük	**Algılayıcı**	**Firma**
Sıcaklık	SHT15 SP30	Sensirion Infineon
Işık	EL7900	Intersil
Nem	SHT15	Sensirion
Basınç	SP30	Infineon
İvme	SP30	Infineon
Gerçek Zamanlı Saat Sıcaklık	DS1629	Dallas

EL7900

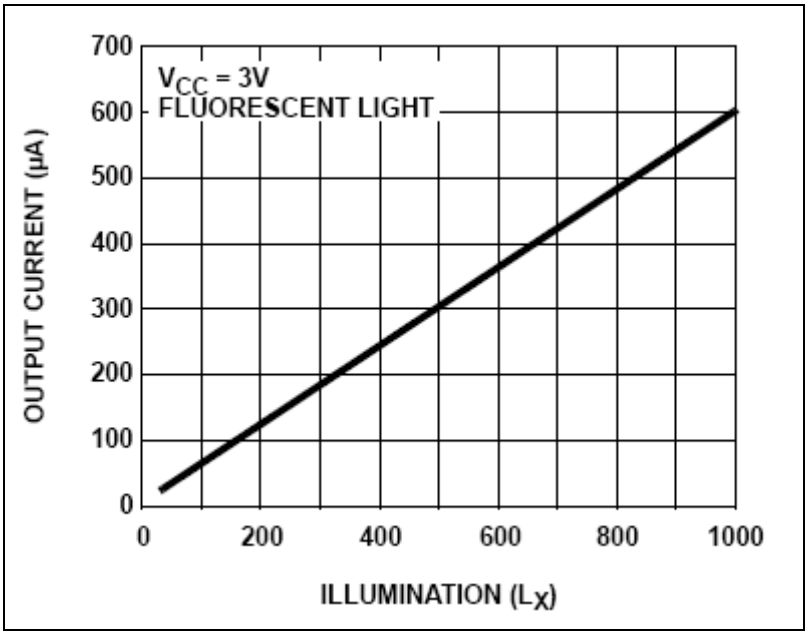
Intersil firması tarafından geliştirilen EL7900 algılayıcısı, çevreden ışık bilgisinin toplanmasında kullanılmaktadır. Algılayıcı, Şekil 6.1'te de görüldüğü gibi bir adet fotodiyot (photodiode) ve transimpedance amplifier dan oluşmaktadır.



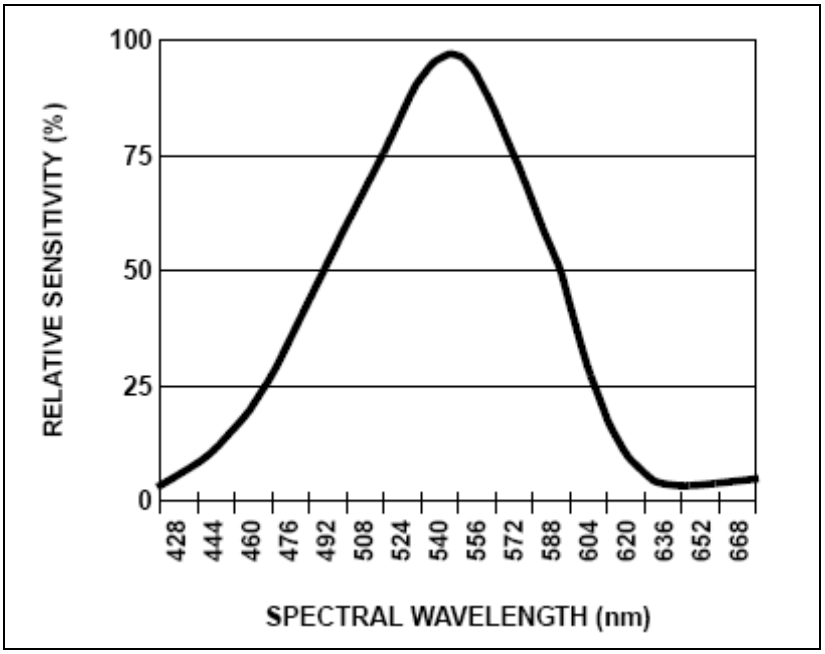
Şekil 6.1 EL7900 iç yapısı ve temel bağlantı biçimi[29]

EL7900'in çıkış geriliminin, ışığın dalga boyuna göre değişimi Şekil 6.2'de, ışığın

parlaklığına göre değişimi ise Şekil 6.3'te grafiksel olarak verilmiştir.



Şekil 6.2 EL7900'ün çıkış geriliminin ışığın parlaklığına göre değişimi[29]



Şekil 6.3 EL7900'ün çıkış geriliminin ışığın dalga boyuna göre değişimi[29]

SHT15

Sensirion firması tarafından geliştirilmiş olan SHT15 algılayıcısı, çevreden sıcaklık ve nem bilgilerini toplayabilmektedir. SHT15 oldukça küçük olan boyutları ve düşük güç tüketimi ile telsiz algılayıcı düğümlerin yapısında kullanılmaya oldukça elverişlidir. Ayrıca Çizelge 6.2'de

de görüldüğü gibi aynı firma tarafından üretilen aynı tipteki algılayıcılara göre hata aralığı çok daha düşüktür. (Tayşi, 2006)

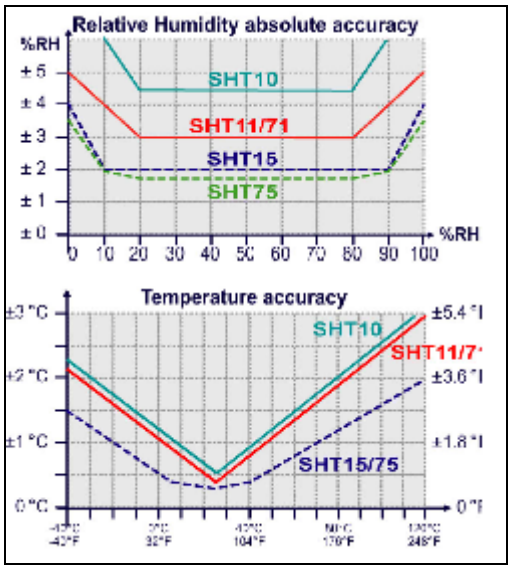
Çizelge 6.2 SHT15'e ait sapma değerleri[27]

Part Number	Humidity accuracy [%RH]	Temperature accuracy [K] @ 25 °C	Package
SHT10	±4.5	±0.5	SMD (LCC)
SHT11	±3.0	±0.4	SMD (LCC)
SHT15	±2.0	±0.3	SMD (LCC)
SHT71	±3.0	±0.4	4-pin single-in-line
SHT75	±1.8	±0.3	4-pin single-in-line

Çizelge 6.3'te SHT15'e ait çeşitli özellikler verilmiştir. Bu özelliklerden bazıları sıcaklık ve nem özelliklerinin ölçüm aralıkları, ölçüm hassasiyetleri ve ölçüm dengeleridir. Şekil 6.4'te ise bu ölçümlerin doğruluk oranlarını gösteren grafikler yer almaktadır.

Çizelge 6.3 SHT15 için nem ve sıcaklık ölçüm özellikleri[27]

Parameter	Conditions	Min.	Typ.	Max.	Units
Humidity					
Resolution (2)		0.5	0.03	0.03	%RH
		8	12	12	bit
Repeatability			±0.1		%RH
Accuracy (1) Uncertainty	linearized	see figure 1			
Range		0		100	%RH
Response time	1/e (63%) slowly moving air		4		s
Temperature					
Resolution (2)		0.04	0.01	0.01	°C
		0.07	0.02	0.02	°F
		12	14	14	bit
Repeatability			±0.1		°C
			±0.2		°F
Accuracy		see figure 1			
Range		-40		123.8	°C
		-40		254.9	°F
Response Time	1/e (63%)	5		30	s



Şekil 6.4 SHT15 sıcaklık ve nem ölçüm doğrulukları[27]

SP30

Infineon firması tarafından geliştirilen SP30 algılayıcısı özellikle arabaların dış tekerlerinde ivme, basınç ve sıcaklık bilgilerinin toplanmasında kullanılmaktadır. Algılayıcı düğüm içerisinde bu 3 fiziksel büyüklüğün çevreden toplanmasında kullanılacaktır. Düğümün yapısında kullanılmak üzere seçilmiş olmasının temel nedenleri; 3 temel fiziksel değeri birarada ölçmesi ve bu fiziksel değerler için oldukça geniş ölçüm aralıklarına sahip olmasıdır. Aşağıdaki çizelgelerde SP30 algılayıcısının sıcaklık, ivme ve basınç ölçüm aralıkları verilmiştir. (Tayşi, 2006)

Çizelge 6.4 SP30 basınç ölçüm aralıkları[28]

Pressure measurement specifications, 100-450kPa range

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Pressure range	100		450	kPa	-40 to 125	
Measurement error	-7		7	kPa	0 to 50	2.1 – 3.6
	-17.5		17.5	kPa	-40 to 125	2.1 – 3.6

Optional pressure measurement ranges

Pressure measurement specifications, 100-700kPa range

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Pressure range	100		700	kPa	-40 to 125	
Measurement error	-11		11	kPa	0 to 50	2.1 – 3.6
	-28		28	kPa	-40 to 125	2.1 – 3.6

Pressure measurement specification, 100-800kPa range

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Pressure range	-100		800	kPa	-40 to 125	
Measurment error	-12,5		12,5	kPa	0 to 50	2.1 - 3.6
	-31.5		31.5	kPa	-40 to 125	2.1 - 3.6

Pressure measurement specifications, 100-900kPa range

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Pressure range	100		900	kPa	-40 to 125	
Measurement error	-14		14	kPa	0 to 50	2.1 – 3.6
	-35		35	kPa	-40 to 125	2.1 – 3.6

Çizelge 6.5 SP30 sıcaklık ölçüm aralığı[28]

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Measurement error	-3		3	°C	-20 to 70	2.1 – 3.6
	-5		5	°C	-40 to 90	2.1 – 3.6
	-3		7	°C	90 to 125	2.1 – 3.6

Çizelge 6.6 SP30 ivme ölçüm aralığı[28]

PARAMETER	SPECIFICATION				AMBIENT CONDITION	
	Min	Typ	Max	Unit	Temp [°C]	VDD [V]
Input range	-12		115	g	-40 to 90	2.1 – 3.6
Sensitivity accuracy	-18.75		18.75	%	-40 to 90	2.1 – 3.6
Offset accuracy	-6		6	g	-20 to 70	2.1 – 3.6
	-8.5		8.5	g	-40 to 90	2.1 – 3.6

DS1629

Dallas Semiconductor firması tarafından üretilen DS1629 algılayıcısı, çevreden sıcaklık bilgisinin okunmasında kullanılmaktadır. Algılayıcı üzerinde ayrıca bir gerçek zamanlı saat bulunmaktadır. Algılayıcı düğümün, mikrodenetleyici birimi üzerinde bulunan sayıcılar yardımı ile belirli periyotlarla çalışması sağlanabilir. Ancak dışardan bağlanacak olan bir gerçek zamanlı saat ile düğümün tam olarak istenen bir tarihte çalışması da sağlanabilir. Mikrodenetleyici biriminde yeralan sayıcılar ile gerçek zamanlı saat kullanılarak elde edilen hassasiyete ulaşmak mümkün değildir. DS1629 üzerinde tarih bilgisinin saklandığı registerlara ait yapı Şekil 5.5'de verilmiştir. Bu yapı ile 2100 yılına kadar olan tarihler tutulabilmektedir.

BYTE ADDRESS	BIT 7 MSb	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 LSb	BYTE RANGE
00h	CH	10 SECONDS			SECONDS				00–59
01h	0	10 MINUTES			MINUTES				00–59
02h	0	12 MODE / 24 MODE	AM/PM / 10 HOURS	10 HOURS	HOURS				01–12 / 00–23
03h	0	0	0	0	0	DAY			01–07
04h	0	0	10 DATE		DATE				01–31*
05h	0	0	0	10 MONTH	MONTH				01–12
06h	10 YEAR				YEAR				00–99

* DATE BYTE MAXIMUM VALUE RANGES FROM 28 TO 31, DEPENDING ON MONTH AND YEAR

Şekil 5.5 DS1629 tarih registerlarının yapısı[30]

7. İŞLETİM SİSTEMİNİN SEÇİMİ VE UYARLANMASI

7.1 Telsiz Algılayıcı Ağ Üzerinde Çalışacak İşletim Sisteminin Seçimi

Günümüzde telsiz algılayıcı düğümler genellikle doğal yaşam alanlarının izlenmesi, tarım ürünlerinin gelişimlerinin izlenmesi, endüstriyel kontrol ve takip sistemleri, erken uyarı sistemleri, kapalı ve açık alanların güvenliği gibi insan erişiminin ve/veya kontolünün zor olduğu durumlarda kullanılmaktadır. Özellikle açık alanlarda kullanılan algılayıcı düğümler, büyük olasılıkla etrafında herhangi bir güç kaynağı olmaması sebebiyle güç ihtiyacını kendi üzerindeki güç kaynağı (piller vs.) tarafından sağlamaktadır. Söz konusu düğümlerin kullanım rahatlığı ve her türlü ortamda kullanılabilmesi için boyutlarının küçük olması gerekmektedir. Bu da, düğümler üzerinde bulunan güç kaynaklarının da küçük olması anlamına gelmektedir. Güç kaynağı küçüldükçe düğümün çalışabileceği zaman da doğru orantılı azalmaktadır. Bu yüzden düğümlerin mümkün olan en düşük güçte çalışması düğümün sorunsuz çalışabileceği süreyi artırmaktadır.

Uyarlanacak işletim sisteminin üzerinde çalışacağı vf-1a isimli telsiz algılayıcı düğüm (Tayşi, Z. Cihan, 2006) tarafından yukarıda bahsi geçen uygulamaların tümünde çalışabilecek şekilde geliştirilmiş genel amaçlı bir düğümdür. Düğümün genel amaçlı olması sebebiyle düğüm üzerinde bulunan ve 6. bölümde detaylı olarak anlatılan donanım elemanları, en düşük güç tüketimiyle en yüksek performansı sağlayabilecek şekilde seçilmiştir. Gücün düğüm üzerinden sağlandığı durumlarda düğümün problemsiz çalışabileceği süreyi artırmak için, üzerinde çalışan işletim sisteminin de düğüm gibi mümkün olan en az enerji ile çalışabilmesi gerekmektedir.

5. bölümde de anlatıldığı üzere algılayıcı düğümler üzerinde olaya dayalı ve çok izlekli olmak üzere iki tür işletim sistemi kullanılmaktadır. Çoklu görevlendirmeli işletim sistemlerinin en yaygın olarak kullanılanı olan MANTIS işletim sisteminin ortalama işlem süresi, olaya dayalı işletim sistemlerinin en yaygın kullanılanı olan TinyOS'a göre daha hızlıdır, fakat TinyOS, MANTIS'e göre bellekte %30 daha az yer kaplamakta ve işletim sistemine binen yükün miktarına göre %5 ile %10 arasında daha az enerji harcamaktadır. İşletim sistemleri arasında yapılan üç karşılaştırmanın ikisinde MANTIS'e göre daha avantajlı olan ve özelllikle MANTIS'e göre daha az enerji harcayan TinyOS işletim sisteminin algılayıcı düğüme uyarlanmasına karar verilmiştir.

7.2 TinyOS İşletim Sisteminin Algılayıcı Düğüme Uyarlanması

TinyOS işletim sisteminin düğüme uyarlanması için beş adım vardır. Bu adımlar:

- **Java 1.5 (Java 5) JDK Yüklenmesi:** Java, bir PC'ye veya laptopa takılı düğüm temel istasyonu veya ağ geçidi ile etkileşim için en uygun yoldur.
- **Yerel Derleyicilerin Yüklenmesi:** Düşük-güçte çalışan mikrodenetleyicilerde derleme işlemi yaparken uygun assembly kodunu üreten derleyiciler gerekir.
- **nesC Derleyicisinin Yüklenmesi:** TinyOS, C dilinin bir türevi olan ve TinyOS eşzamanlı kullanım modelini ve bileşene dayalı programlamayı destekleyen nesC dilinde yazılmıştır. nesC dili platformdan bağımsızdır.
- **TinyOS Kaynak Ağacının Yüklenmesi:** TinyOS'un derlenip yüklenebilmesi için koda ihtiyaç duyulmaktadır.
- **Graphviz Görsel Aracın Yüklenmesi:** TinyOS ortamı kaynak koddan HTML dokümantasyonu oluşturan *nesdoc* aracını içerir.Bu işlemin bir kısmı farklı TinyOS bileşenleri arasındaki ilişkiyi gösteren diyagramları çizmektir. Graphviz, *nesdoc* un kullandığı diyagramları çizmeye yarayan açık kaynak kodlu bir araçtır.

7.2.1 Java 1.5 JDK Yüklenmesi

Java 1.5 JDK IBM'in web sitesinden adresinden indirilip yüklenmiştir.

7.2.2 Yerel Derleyicilerin Yüklenmesi

Texas Instrument MSP430 araçları TinyOS'un web sitesinden indirilmiş ve aşağıdaki sıra ile yüklenmiştir.

Temel araçlar: rpm –ivh msp430tools-base-0.1-20050607.i386.rpm

python araçları: rpm –ivh msp430tools-python-tools-1.0-1.noarch.rpm

binutils: rpm –ivh msp430tools-binutils-2.16-20050607.i386.rpm

gcc: rpm –ivh msp430tools-gcc-3.2.3-20050607.i386.rpm

libc: rpm –ivh msp430tools-libc-20050308cvs-20050608.i386.rpm

jtag: rpm –ivh msp430tools-jtag-lib-20031101cvs-20050610.i386.rpm

gdb: rpm –ivh msp430tools-gdb-6.0-20050609.i386.rpm

7.2.3 nesC Derleyicisinin Yüklenmesi

TinyOS'a özel araçlar NesC derleyicisi ve *tinyos-2.x/tools* kaynak kod veri havuzunda geliştirilen araçlardır. Bu araçlar da *rpm* ler kullanılarak yüklenirler. NesC ve TinyOS araçları aşağıdaki komutlar kullanılarak yüklenmişlerdir.

rpm –Uvh nesc-1.2.8a-1.i386.rpm

rpm –Uvh tinyos-tools-1.2.4-3.i386.rpm

7.2.4 TinyOS 2.x Kaynak Ağacının Yüklenmesi

Araçlar yüklendikten sonra TinyOS 2.x kaynak ağacının yüklenmesi ve ortam değişkenlerinin ayarlanması gerekir. TinyOS 2.x kaynak ağacı aşağıdaki komut kullanılarak yüklenmiştir.

rpm –ivh tinyos-2.0.2-2.noarch.rpm

TnyOS 2.x kaynak ağacı yüklendikten sonra ortam değişkenlerinin de ayarlanması gerekir. Kolaylık olması amacıyla ortam değişkenleri bir kabuk betik (shell script) konmuştur. Çizelgede 7.1 hangi ortam değişkenlerinin nerede tutulduğunu gösterir.

Çizelge 7.1 TinyOS 2.x ortam değişkenlerinin tutulduğu alan

Environment Variable	**Linux**
TOSROOT	/opt/tinyos-2.x
TOSDIR	$TOSROOT/tos
CLASSPATH	$TOSROOT/support/sdk/java/tinyos.jar:.
MAKERULES	$TOSROOT/support/make/Makerules
PATH	/opt/msp430/bin:$PATH

7.2.5 Graphviz'in Yüklenmesi

TinyOS yüklemesinin son adımı olarak Graphviz'in web sitesinden Graphviz indirilmiş ve aşağıdaki komut ile yüklenmiştir.

rpm – i graphviz-2.20.2-1.src.rpm

7.3 TinyOS Üzerinde vf-1a Algılayıcı Düğüm Platformunun Oluşturulması

TinyOS işletim sistemi içerisinde, üzerinde çalıştığı algılayıcı ağın düzgün çalışabilmesini sağlamak amacıyla yazılan konfigürasyon dosyalarının ve algılayıcıların sürücü dosyalarının bulunduğu bir platform yaratılması gerekmektedir. Platform yaratma işleminin ilk adımı *tos/platforms* dizini altına telsiz algılayıcı ağın ismi ile bir dizin oluşturmaktır.[40]

7.3.1 .platform Dosyası

Her platform dizini içerisinde, platformun derleyici parametre bilgilerini bulunduran *.platform* isimli bir dosya bulunmalıdır. Bu yüzden *tos/platforms/vf-1a/.platform* dosyası oluşturulmuş ve dosyanın içeriği aşağıdaki gibi düzenlenmiştir.

```
push( @includes, qw(
  %T/platforms/vf-1a
  %T/platforms/vf-1a/chips/cc2420
  %T/platforms/vf-1a/chips/stm25p
  %T/platforms/vf-1a/chips/ds1629
  %T/chips/cc2420
  %T/chips/cc2420/alarm
  %T/chips/cc2420/control
  %T/chips/cc2420/csma
  %T/chips/cc2420/interfaces
  %T/chips/cc2420/link
  %T/chips/cc2420/lowpan
  %T/chips/cc2420/lpl
  %T/chips/cc2420/packet
  %T/chips/cc2420/receive
  %T/chips/cc2420/spi
  %T/chips/cc2420/transmit
  %T/chips/cc2420/unique
  %T/chips/msp430
  %T/chips/msp430/adc12
  %T/chips/msp430/pins
  %T/chips/msp430/timer
  %T/chips/msp430/usart
  %T/chips/msp430/sensors
  %T/chips/stm25p
  %T/lib/timer
  %T/lib/serial
  %T/lib/adc
  %T/lib/power
) );

@opts = qw(
  -gcc=msp430-gcc
  -mmcu=msp430x149
  -fnesc-target=msp430
```

```
  -fnesc-no-debug
  -fnesc-
scheduler=TinySchedulerC,TinySchedulerC.TaskBasic,TaskBasic,TaskBasic,runTask,postTa
sk
);
push @opts, "-mingw-gcc" if $cygwin;
```

Bu dosya ncc derleyicisi tarafından yorumlanan perl parçacıkları içerir. İlk ifade herhangi bir uygulama vf-1a platformu için derlenirken gerekli olan sürücü veya diğer dosyaların bulunduğu dizinleri ekler (%T *tinyos-2.x/tos* dizinini gösteren çevresel değişkendir).

İkinci ifade, necs'ye aktarılan çeşitli parametreleri içeren *@opts* listesini tanımlar.[40]

7.3.2 hardware.h Dosyası

Her platform dizininde; platforma özel sabitleri, pin numaralarını ve diğer harici başlık dosyalarını (örneğin *msp430hardware.h*) içeren bir *hardware.h* dosyası olmalıdır. Bu yüzden */tos/platforms/vf-1a/hardware.h* dosyası aşağıdaki bilgileri içerecek şekilde oluşturulmuştur:

```
#ifndef _H_hardware_h
  #define _H_hardware_h

  #include "msp430hardware.h"

  // LEDs
  TOSH_ASSIGN_PIN(RED_LED, 5, 4);
  TOSH_ASSIGN_PIN(GREEN_LED, 5, 5);
  TOSH_ASSIGN_PIN(YELLOW_LED, 5, 6);

  // CC2420 RADIO #defines
  TOSH_ASSIGN_PIN(RADIO_CSN, 4, 1);
  TOSH_ASSIGN_PIN(RADIO_VREF, 4, 2);
  TOSH_ASSIGN_PIN(RADIO_RESET, 4, 3);
  TOSH_ASSIGN_PIN(RADIO_FIFOP, 1, 0);
  TOSH_ASSIGN_PIN(RADIO_SFD, 4, 0);
  TOSH_ASSIGN_PIN(RADIO_GIO0, 1, 1);
  TOSH_ASSIGN_PIN(RADIO_FIFO, 1, 1);
  TOSH_ASSIGN_PIN(RADIO_GIO1, 1, 2);
  TOSH_ASSIGN_PIN(RADIO_CCA, 1, 2);
  TOSH_ASSIGN_PIN(CC_FIFOP, 1, 0);
  TOSH_ASSIGN_PIN(CC_FIFO, 1, 1);
  TOSH_ASSIGN_PIN(CC_SFD, 4, 0);
  TOSH_ASSIGN_PIN(CC_VREN, 4, 2);
  TOSH_ASSIGN_PIN(CC_RSTN, 4, 3);

  // UART pins
  TOSH_ASSIGN_PIN(SOMI0, 3, 2);
  TOSH_ASSIGN_PIN(SIMO0, 3, 1);
```

```
TOSH_ASSIGN_PIN(UCLK0, 3, 3);
TOSH_ASSIGN_PIN(UTXD0, 3, 4);
TOSH_ASSIGN_PIN(URXD0, 3, 5);
TOSH_ASSIGN_PIN(UTXD1, 3, 6);
TOSH_ASSIGN_PIN(URXD1, 3, 7);
TOSH_ASSIGN_PIN(UCLK1, 5, 3);
TOSH_ASSIGN_PIN(SOMI1, 5, 2);
TOSH_ASSIGN_PIN(SIMO1, 5, 1);

// DS1629 pins
TOSH_ASSIGN_PIN(DS1629_SCL, 1, 3);
TOSH_ASSIGN_PIN(DS1629_SDA, 1, 4);
TOSH_ASSIGN_PIN(DS1629_RTC_ALRM, 1, 5);

#endif // _H_hardware_h
```

Bu dosya *.platform* dosyasında tanuımladığımız */tos/chips/msp430* dizininde bulunan *msp430hardware.h* dosyasını çeker ve bu dosyada bulunan makroları kullanarak bazı fiziksel pinlerin tanımlamalarını yapar. Örneğin vf-1a düğümünde kırmızı led fiziksel olarak genel amaçlı G/Ç 5.4 pinine bağlıdır. [40]

7.3.3 vf-1a İçin make Tanımlama

Herhangi bir uygulama bir platform altında çalıştırılmak istendiğinde, TinyOS derleme sisteminin bu platformu tanıması gerekir. TinyOS derleme sistemi *tinyos-2.x/support/make* dizininde bulunur. Bir platformun derleme sistemi tarafından tanınması için ise belirtilen dizinin içerisinde *platform_adı.target* isimli bir dosyanın oluşturulması gerekir. Bu yüzden */tinyos-2.x/support/make/vf-1a.target* dosyası oluşturulmuş ve içeriği aşağıdaki gibi düzenlenmiştir. [40]

```
PLATFORM = vf-1a
MSP_MCU = msp430x149
# Disable MSP430 hardware multiply because it makes MSPGCC die

PFLAGS += -mdisable-hwmul
OPTFLAGS += -O
MSP_BSL ?= tos-bsl
ifdef CC2420_CHANNEL
PFLAGS += -DCC2420_DEF_CHANNEL=$(CC2420_CHANNEL)
endif

$(call TOSMake_include_platform,msp)

vf-1a: $(BUILD_DEPS)
    @:
```

7.3.4 PlatformC.nc ve PlatformP.nc Dosyaları

Yeni bir platforma taşınan TinyOS bağlantısı, *PlatformC* isimli ve o platforma ait ve sadece *Init* arayüzü içeren bir bileşene sahip olmalıdır. Bu arayüzü gerçekleştirmek için *tos/platforms/vf-1a/PlatformP.nc* dosyası;

```
#include "hardware.h"

module PlatformP{
  provides interface Init;
  uses interface Init as Msp430ClockInit;
  uses interface Init as LedsInit;
}
implementation {
  command error_t Init.init() {
    call Msp430ClockInit.init();
    call LedsInit.init();
    return SUCCESS;
  }

  default command error_t LedsInit.init() { return SUCCESS; }

}
```

ve tos/platforms/vf-1a/PlatformC.nc dosyaları oluşturulmuştur.

```
#include "hardware.h"

configuration PlatformC {
  provides interface Init;
}
implementation {
  components PlatformP
    , Msp430ClockC
    ;

  Init = PlatformP;
  PlatformP.Msp430ClockInit -> Msp430ClockC.Init;
}
```

7.3.5 PlatformLedsC.nc Dosyası

Ledlerin herhangi bir platform üzerinde çalışabilmesi için *PlatformLedsC* bileşenine ihtiyaç duyulmaktadır. Bu bileşen platforma özel bir bileşen olduğu için vf-1a platformu için tanımlamasının yapılması gerekmektedir. Ledlerin platforma özel bir bileşene ihtiyaç duyması donanım seviyesinde her platform için farklı gerçekleştirilmiş olmalarındandır. Ledler mikrodenetleyicinin G/Ç pinlerinden birine bağlıdır ve açma/kapama işlemi ilgili pinin çıkışına 0/1 vererek sağlanır. Fakat bu basit ve tekbiçimli modelde bile Ledlerin açma/kapama

işleminin tanımlanması için platforma özel bir en alt seviye vardır.

vf-1a üzerinde bulunan üç Led için gereken üç *GeneralIO* arayüzü ve *Init* içeren *PlatformLedsC* konfigürasyonu *tos/platforms/yamp/PlatformLedsC.nc* dosyasında aşağıdaki gibi tanımlanmıştır.

```
#include "hardware.h"

configuration PlatformLedsC {
  provides interface GeneralIO as Led0;
  provides interface GeneralIO as Led1;
  provides interface GeneralIO as Led2;
  uses interface Init;
}
implementation {
  components
     HplMsp430GeneralIOC as GeneralIOC
   , new Msp430GpioC() as Led0Impl
   , new Msp430GpioC() as Led1Impl
   , new Msp430GpioC() as Led2Impl
   ;
  components PlatformP;

  Init = PlatformP.LedsInit;

  Led0 = Led0Impl;
  Led0Impl -> GeneralIOC.Port54;

  Led1 = Led1Impl;
  Led1Impl -> GeneralIOC.Port55;

  Led2 = Led2Impl;
  Led2Impl -> GeneralIOC.Port56;
}
```

7.3.6 vf-1a Düğümü Üzerindeki Algılayıcıların Sürücü Dosyaları

Bu dosya da oluşturulduktan sonra vf-1a platformu, üzerinde Ledleri yakıp söndürme gibi basit uygulamalar çalıştırılacak duruma gelmiştir. Fakat vf-1a düğümü üzerinde bulunan algılayıcıların uygulamalar içerisinde kullanılabilmeleri için sürücü dosyalarının vf-1a platformuna uygun şekilde yazılmaları gerekir.

vf-1a düğümü üzerinde bulunan CC2420 telsiz iletişim biriminin sürücü dosyaları TinyOS kurulumu ile birlikte gelen *telosa* platformu içerisinde mevcuttur. *telosa* düğümünde de MSP430 mikroişlemcisi olmasından dolayı vf-1a platformu altında da bu sürücüyü kullanmak mümkündür. Bunun için *tos/platforms/telosa/chips/cc2420* dizini *tos/platforms/vf-1a/chips* dizini altına kopyalanmıştır.

Düğüm üzerinde olan M25P80 ikincil bellek biriminin düzgün çalışması için gereken sürücü dosyaları TinyOS kurulumu ile birlikte gelen *telosb* platformu içerisinde mevcuttur. *telosb* düğümünde de MSP430 mikroişlemcisi olmasından dolayı vf-1a platformu altında da bu sürücüyü kullanmak mümkündür. Bunun için *tos/platforms/telosb/chips/stm25p* dizini *tos/platforms/vf-1a/chips* dizini altına kopyalanmıştır.

Üzerinde gerçek zamanlı saat de bulunan DS1629 ısı algılayıcısı, mikroişlemci ile I^2C protokolünü kullanarak haberleşmektedir. I^2C protokolü her bir *clock* zamanında 1 bit'lik bilgi gönderilmesini veya alınmasını sağlar. Bu bit'ler 1 byte'lık paketler halinde komut ya da adres olarak gönderilirler. DS1629 algılayıcısının sıcaklık ölçme, o anki tarih bilgisi okuma gibi komutları sabittir ve mikroişlemci bu komutlardan herhangi birisini algılayıcıya gönderdiği zaman algılayıcı bu komutu gerçekleştirir. Algılayıcı üzerinden okuma/yazma yapılmak istendiğinde mikroişlemcinin hangi komutu algılayıcıya göndermesi gerektiğini ve algılayıcıya hangi pinleri üzerinden bağlı olduğunu bilmesi için bir başlık dosyası oluşturulması gerekmektedir. Bu yüzden *tos/platforms/vf-1a/chips/ds1629/DS1629.h* dosyası aşağıdaki gibi oluşturulmuştur.

```
#ifndef _DS1629_H
#define _DS1629_H

#define DS1629_PTR_ACCONF          (0xAC)
#define DS1629_PTR_ACMEM           (0x17)
#define DS1629_PTR_STRTCONVTEMP    (0xEE)
#define DS1629_PTR_STPCONVTEMP     (0x22)
#define DS1629_PTR_READTEMP        (0xAA)
#define DS1629_PTR_READCNTR        (0xA8)
#define DS1629_PTR_READSLP         (0xA9)
#define DS1629_PTR_ACTH            (0xA1)
#define DS1629_PTR_ACTL            (0xA2)
#define DS1629_PTR_ACCL            (0xC0)
#define DS1629_PTR_ACCLAL          (0xC7)

/*#define DS1629_SC_PORF             (1 << 6)
#define DS1629_SC_SMOD        (1 << 5)
#define DS1629_SC_NBEN        (1 << 4)
#define DS1629_SC_PIO              (1 << 3)
#define DS1629_SC_FQ(_x)      (((_x) & 0x3))*/

#endif /* _DS1629_H */
```

Başlık dosyası oluşturulduktan sonra sürücünün kullanılabilir hale gelmesi için HPL (Hardware Presentation Layer) arayüzünün hazırlanması gerekir. Bu arayüz, herhangi bir uygulama geliştirmek istendiğinde, algılayıcıya erişmek için gerekli komut byte'ını

bilmemize gerek kalmadan olayları (event) kullanmamıza imkan sağlar. Aşağıda DS1629 algılayıcısı için hazırlanmış sürücü dosyalarından sırasıyla *HplDS1629.nc* ve *HplDS1629LogicP.nc* dosyalarında bulunan olaylara ait örnek verilmiştir.

```
/**
   * Reads last temperature conversion result.
   *
   * @return SUCCESS if the conversion will be made
   */
  command error_t readTemperature();

  /**
   * Presents the result of a temperature conversion.
   *
   * @param error SUCCESS if the reading was successful
   * @param val the temperature reading
   */
  async event void readTemperatureDone( error_t error, uint16_t val );


implementation {
  enum {
    STATE_READTEMP,
  }
  command error_t HplDS1629.readTemperature() {
    return doReadReg(STATE_READTEMP,DS1629_PTR_READTEMP);
  }
  default async event void HplDS1629.readTemperatureDone( error_t error, uint16_t val ){
return; }
}
```

8. İŞLETİM SİSTEMİNİ DÜĞÜM ÜZERİNDE TEST EDEN ÖRNEK UYGULAMA

TinyOS işletim sistemi vf-1a telsiz algılayıcı düğümü üzerine uyarlandıktan ve düğümün işletim sistemi altında çalışabilmesi için gerekli sürücü dosyaları oluşturulduktan sonra düğümün çalışmasını test etmek amacıyla bir örnek uygulama yazılmıştır. Bu örnek uygulama ile, ortamdan ısı algılayıcısı vasıtasıyla sıcaklık bilgisi alınmış ve sıcaklığın şiddetine göre düğüm üzerinde bulunan Led'ler açılıp kapatılmıştır.

Oluşturulan uygulamalar, TinyOS işletim sisteminde *apps/* dizini altında bulunmaktadırlar. Bu yüzden aynı zamanda uygulama ileaynı isme sahip olan TemperatureToLeds dizini belirtilen dizin altında oluşturulmuştur. Her uygulamanın bir *Makefile* dosyasına sahip olması gerekmektedir, bu yuzden bu dizin altında *Makefile* dosyası aşağıdaki içerie sahip olacak şekilde oluşturulmuştur.

```
COMPONENT=TemperatureToLedsAppC
include $(MAKERULES)
```

Makefile içerisinde bileşen (component) olarak belirtilen *TemperatureToLedsAppC* bu uygulamanın çalışabilmesi için gereken diğer bileşenlerle arada bağlantıların kurulmasını sağlayan bir konfigürasyon dosyasıdır. Led'lerin açılıp kapanması için *LedsC*, belirli aralıklarla sıcaklık bilgisinin okunmasının sağlanabilmesi için *TimerMilliC* ve okuma işleminin DS1629 algılayıcısı üzerinden yapılabilmesi için *DS1629*bileşenlerine ihtiyaç vardır. *TemperatureToLedsC* uygulamasının bu bileşenler ile bağlantısı aşağıdaki gibi yapılmıştır.

```
configuration TemperatureToLedsAppC {}
implementation {
  components MainC, TemperatureToLedsC as App, LedsC;
  components new TimerMilliC();
  components new DS1629();

  App.Boot -> MainC.Boot;
  App.ReadTemp -> DS1629;
  App.Leds -> LedsC;
  App.MilliTimer -> TimerMilliC;
}
```

Gerekli bağlantılar yapıldıktan sonra geriye sadece uygulamanın yazılması kalmıştır. Bu uygulama ortamın sıcaklığını 250ms'lik aralıklarla okur. Eğer sıcaklık 20˚C altında ise sadece yeşil Led'in yanmasını, sıcaklık 20˚C ile 30˚C arasında ise sarı Led'in yanmasını, sıcaklık 30˚C üzerinde ise kırmızı Led'in yanmasını, algılayıcıdan okuma esnasında bir hata oluştu ise üç Led'in birden yanmasını sağlar. Bu işlemlerin yapılmasını sağlayan

TemperatureToLedsC.nc dosyasının içeriği aşağıda gösterilmiştir.

```
#include "Timer.h"
#include "TemperatureToLeds.h"

module TemperatureToLedsC {
  uses {
    interface Leds;
    interface Boot;

    interface Timer<TMilli> as MilliTimer;
          interface SplitControl as Control;
          interface ReadTemperature<uint16_t> as ReadTemp;

  }
}
implementation {

  uint16_t counter = 0;

  static void fatal_problem();
  static void report_problem();

  event void Boot.booted() {
          call Control.start();
          call MilliTimer.startPeriodic(250);
  }


  event void Control.startDone(error_t err) {
    if (err == SUCCESS) {
//
    }
    else {
      call Control.start();
    }
  }

  event void Control.stopDone(error_t err) {
    // do nothing
  }

  event void MilliTimer.fired() {

          call MilliTimer.startPeriodic(250);

          if (call ReadTemp.read() != SUCCESS)
      fatal_problem();
  }
```

```
  event void ReadTemp.readDone(error_t result, uint16_t temp) {

    if (result != SUCCESS) {
      temp = 0xffff;
      report_problem();
    }
          if ((temp < 20) || (temp > 32767)) {
                call Leds.led0On();
                call Leds.led1Off();
                call Leds.led2Off();
          }
          else if ((temp >= 20) && (temp < 30)) {
             call Leds.led0Off();
                call Leds.led1On();
                call Leds.led2Off();
          }
          else if ((temp >= 30) && (temp <= 32767)) {
             call Leds.led0Off();
                call Leds.led1Off();
                call Leds.led2On();
          }

  }

  static void report_problem() {
    call Leds.led0Toggle();
  }

  // Use LEDs to report various status issues.
  static void fatal_problem() {
    call Leds.led0On();
    call Leds.led1On();
    call Leds.led2On();
    call MilliTimer.stop();
  }
}
```

9. SONUÇ

Doğal yaşam alanlarının izlenmesi, tarım ürünlerinin gelişimlerinin izlenmesi, endüstriyel kontrol ve takip sistemleri, erken uyarı sistemleri, kapalı ve açık alanların güvenliğinin sağlanması ve daha pek çok amaçla kullanılabilen telsiz algılayıcı ağlar oldukça güncel bir konu olmasına rağmen kullanımı ülkemizde yeterince yaygınlaşamamıştır. Bu tez çalışmasında hedeflenen, var olan bir telsiz algılayıcı düğümün üzerine yine var olan bir işletim sistemini uyarlamaktır.

İşletim sisteminin uyarlanacağı düğüm yukarıda bahsi geçen uygulamaların her birinde çalışabilecek şekilde geliştirilmiş genel amaçlı bir düğümdür. Genel amaçlı düğümlerin isminden de anlaşılabileceği üzere günlük hayatta her türlü alanda çalışabilmesi gerekmektedir. Fakat doğal yaşam alanlarının izlenmesi gibi durumlarda düğümleri her yerde dışarıdan bir güç kaynağı aracılığıyla besleme imkanı yoktur, bu da düğümün kendi kendini besleyebilmesi gerekliliğini ortaya koymuştur. Boyut olarak santimetreler, milimetreler mertebesine kadar küçülen düğümler üzerinde enerji ihtiyacını uzun süre karşılayabilmek pek mümkün olamamaktadır. Bu yüzden düğümün daha uzun süre çalışabilmesi için düğüm üzerinde bulunan donanım ve yazılım elemanlarının mümkün olan en az enerji ile çalışmasını sağlamak gerekmektedir. İşletim sisteminin uyarlanacağı düğüm üzerinde bulunan donanım elemanları bu özelliği sağlamaktadır. Düğüm üzerinde uyarlanacak işletim sisteminin de bu yönde seçilmesi, düğümün çalışma süresini artıracaktır. Bu yüzden işletim sistemi seçilirken bu kriter ön plana alınmıştır ve enerjiyi en efektif kullanan olaya dayalı işletim sistemlerinden en yaygın kullanılanı olan TinyOS işletim sistemi kullanılmıştır.

Donanım seviyesine daha yakın olan nesC'nin bileşene dayalı bir yapısının olması, sürücüleri oluşturmanın ve uygulama geliştirmenin beklenilenden daha zor olmasına ve bu yüzden uygulama geliştirmenin uzun zaman almasına sebebiyet vermiştir. Özellikle TinyOS ortamında ilk defa uygulama geliştirmek isteyen kişiler için bu durum önemli bir problem teşkil etmektedir.

KAYNAKLAR

Bhatti, S., Carlson, J., Dai, H., Deng, J., Rose, J., Sheth, A., Shucker, B., Gruenwald, C. , Torgerson, A., Han, R., (2005) "MANTIS OS: An Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms", ACM/Kluwer Mobile Networks & Applications (MONET), Special Issue on Wireless Sensor Networks, vol. 10, no. 4, Ağustos 2005

Chong C., Kumar S.P., (2003) "Sensor Networks : Evolution, Opportunities, and Challenges", Proceedings of IEEE Volume 91 Issue 8, Ağustos 2003

Cormac Duffy, Utz Roedig, John Herbert, Cormac Sreenan, (2007), "An Experimental Comparison of Event Driven and Multi-Threaded Sensor Node Operating Systems" Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops 2007

Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., Anderson, J., (2002) "Wireless sensor networks for habitat monitoring", Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications, WSNA 2002, Atlanta, Georgia, USA, Eylül 2002

Polastre, J., Szewczyk, R., Culler, D., "Telos: Enabling Ultra-Low Power Wireless Research", The Fourth International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors (IPSN/SPOTS), Los Angeles, USA, Nisan 2005.

Szewczyk, R., Polastre, J., Mainwaring, A.M., Culler, D., (2004) "Lessons from a Sensor Network Expedition" Proceedings of Wireless Sensor Networks, First European Workshop, EWSN 2004, Berlin, Germany, Ocak 2004

Tayşi, Z. Cihan (2006) "Genel Amaçlı Telsiz Algılayıcı Bir Düğümün Tasarımı ve Gerçeklenmesi".

Zhang, P., Sadler, C., Lyon, S., Martonosi, M., "Hardware Design Experiences in ZebraNet", The Second ACM Conference on Embedded Networked Sensor Systems. Kasım 2004

INTERNET KAYNAKLARI

[1]www.chinfo.navy.mil/navpalib/cno/n87/usw/issue_25/sosus.htm

[2]www.greatduckisland.net

[3]www.princeton.edu/~mrm/zebranet.html

[4]www.darpa.mil/body/archives/sensit/index.htm

[5] Cebrowski, A.K., Garstka, J.J., "Network-Centric Warfare: Its Origin and Future"**,** www.usni.org/Proceedings/Articles98/PROcebrowski.htm

[6]robotics.eecs.berkeley.edu/~pister/29Palms0103/

[7]robotics.eecs.berkeley.edu/~pister/SmartDust/

[8]Bonet, P., "The Hogthrob Project",

"www.hogthrob.dk/output/publications_files/Hogthrob-Zurich.ppt"

[9]AtMega 128L Data sheet", **"**www.atmel.com/atmel/acrobat/doc2467.pdf ",

[10]www.eecs.harvard.edu/~mdw/proj/codeblue/

[11]Kang, K.D., "Introduction to TinyOS",

"www.cs.binghamton.edu/~kang/cs580s/tinyos.ppt",

[12]MSP430S149 Data sheet, "focus.ti.com/lit/ds/symlink/msp430f149.pdf", revised June 2004

[13]AT45DB041B Data sheet",

"www.atmel.com/dyn/resources/prod_documents/doc3443.pdf",

[14]TR1000 Data sheet, "www.rfm.com/products/data/tr1000.pdf"

[15]CC2420 Data sheet, "www-inst.eecs.berkeley.edu/~cs150/Documents/CC2420.pdf"

[16]www.eng.yale.edu/enalab/XYZ/

[17]OKIML67Q500x Datasheet,

"www.eng.yale.edu/enalab/XYZ/programming/oki_manual.pdf"

[18]www.acm.uiuc.edu/sigarch/projects/sensornode/

[19]www.eyes.eu.org/sensnet.htm

[20]www.moteiv.com/

[21] MicaZ Data sheet,

"www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Kit_Datasheet.pdf"

[22]Hill, J., "A Software Architecture Supporting Networked Sensors",

"www.cs.berkeley.edu/~kwright/nest_papers/TinyOS_Masters.pdf",

[23]MANTIS Project homepage, "mantis.cs.colorado.edu/index.php/tiki-index.php",

[24]SOS Project homepage, “nesl.ee.ucla.edu/projects/sos/”

[25]AVRx homepage, “www.barello.net/avrx/index.htm”

[26]M25P80, “www.st.com/stonline/books/pdf/docs/8495.pdf”

[27]SHT15 Datasheet, “www.sensirion.com/images/getFile?id=25”

[28]SP30 Datasheet, “www.infineon.com/upload/Document/SP30%20product%20brief.pdf”

[29]EL7900 Datasheet, “http://www.intersil.com/data/fn/fn7377.pdf”

[30]DS1629 Datasheet, “www.maxim-ic.com/quick_view2.cfm/qv_pk/2739”

[31]www.cs.berkeley.edu/~kamin/calamari/

[32]firebug.sourceforge.net/

[33]www-bsac.eecs.berkeley.edu/projects/cotsbots/

[34]telegraph.cs.berkeley.edu/tinydb/

[35]ptolemy.eecs.berkeley.edu/papers/03/TinyGALS

[36]www.tendrilinc.com

[37]firecenter.forestry.umt.edu/firecenter/

[38]www.tinyos.net/tinyos-2.x/doc/html/install-tinyos.html

[39]docs.tinyos.net/index.php/Mote-mote_radio_communication

[40]http://docs.tinyos.net/index.php/Platforms

ÖZGEÇMİŞ

Doğum tarihi	22.09.1979	
Doğum yeri	İstanbul	
Lise	1989-1996	Cağaloğlu Anadolu Lisesi
Lisans	1996-2003	Yıldız Teknik Üniversitesi Elektrik Elektronik Fak. Bilgisayar Mühendisliği Bölümü

Çalıştığı kurum(lar)

	2003-2009	YTÜ Bilgi İşlem Daire Başkanlığı