IDLE ve POWER_DOWN Güç Durumunun Enerji Başarım Değerlendirmesi

Sinan TOKLU[1], O. Ayhan ERDEM[2]

[1]Gazi Üniversitesi, Bilişim Enstitüsü, Elektronik-Bilgisayar Eğitimi Anabilim Dalı, Teknikokullar, ANKARA
[2]Gazi Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Bölümü, Teknikokullar, ANKARA
s.toklu@gazi.edu.tr, ayerdem@gazi.edu.tr
(Geliş/Received:19.07.2012; Kabul/Accepted:12.11.2012)

Özet— Kablosuz Algılayıcı Ağlar (KAA), araştırmacılar için ilgi çekici bir araştırma alanı olmuştur. Kablosuz Algılayıcı düğümlerin düşük işlemci, düşük bellek, sınırlı enerji gibi problemleri bulunmaktadır. Fakat rastgele yerleştirilebilme, kendi kendine örgütleyebilme, beraber çalışma ve yerel hesaplama yapma gibi özelliklere sahip olan algılayıcı düğümler; askeri, çevresel, sağlık ve ticari olmak üzere birçok alanda kullanılmaktadır. Kablosuz algılayıcı ağlardaki en büyük sorun ise enerji verimliliği olarak karşımıza çıkmaktadır. Bu alanda birçok çalışma yapılmıştır. Bu makalede genel olarak KAA enerji tüketimini azaltmaya yönelik güç durumlarından bahsedilmekte ve TinyOS 1.x işletim sistemi yardımıyla POWER_DOWN ve IDLE kipi güç durumları temel alınarak, bu durumların enerji tüketimlerinin karşılaştırmalı olarak performans analizi sunulmaktadır.

Anahtar Kelimeler— Kablosuz algılayıcı ağlar, enerji-verimli, güç yöntemleri, Tinyos, güç kapama, boşta durma

Performance Evaluation of IDLE and POWER_DOWN Power Mode

Abstract— Wireless Sensor Networks (WSNs) have become an interesting research area for the researchers. Wireless sensor nodes have some problems such as low processor, low memory and limited energy. However, sensor nodes are used for several important tasks by means of some abilities of nodes, such as self organizing, collaboration, local computation. Therefore, sensor networks can be applied several areas such as military, environmental, health and commercial areas. The most important problem of wireless sensor network is energy efficiency. In this paper, energy saving methods in TinyOS are explained in general and comparative performance analysis is presented on POWER_DOWN and IDLE power mode with the aid of TinyOS 1.x program.

Keywords— Wireless sensor networks, energy-efficient, power mode, Tinyos, POWER_DOWN, IDLE

1.GİRİŞ

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

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

Algılayıcı ağlardaki amaç, ağı oluşturan düğümlerin ucuz ve güç kaynaklarının verimli olarak kullanılmasıdır. Ağda bulunan düğümlerin bazıları veya tamamı pillerinin bitmesi nedeniyle veya konum değiştirme nedenleri gibi olaylardan dolayı ağdaki ömürlerini bitirirler. Bu da zamanla ağın ömrünü azaltmaktadır [1].

Bu çalışmada TinyOS işletim sistemi kullanılarak MICA2 düğümlerde bulunan güç durumlarının başarım analizi yapılmıştır. MICA2 düğümleri için TinyOS1.x birçok güç durumu sunmaktadır. Bu güç durumları kullanılarak düğümün enerji seviyesi CPU bazında düşürülmektedir. Buna ek olarak Güç-Durumu bileşeni kullanılarak çevre birimleri örneğin radyo ve LED gibi bileşenler uygulama içinde dinamik olarak kapatılıp açılarak enerji verimliliği sağlanmaktadır.

Enerji tüketiminin en önemli faktörlerinden birisi olan istem dışı alım düğümlerin çalışmadığı zamanda TinyOS 1.x'in sunduğu güç durumları ve Güç-Durumu bileşeni kullanılarak en düşük seviyeye indirilebilmektedir.

Bu çalışmada TinyOS işletim sisteminde bulunan PowerTOSSIM yapısı ve daha sonra ise ilgili güç durumlarından IDLE ve POWER_DOWN güç yönetimi açıklanacaktır.

2. PowerTOSSIM

PowerTOSSIM kablosuz algılayıcı ağlar için ölçeklenebilir bir ortam ve her düğüm için kesine yakın enerji tüketimi tahmini sunmaktadır. PowerTOSSIM, TOSSIM'in genişletilmiş halidir ve TinyOS uygulamaları için olaya dayalı benzetim ortamı sunmaktadır. PowerTOSSIM'de TinyOS bileşenleri ile ilgili olarak belirli donanım durumları için (örneğin radyo, EEPROM, LEDs, v.s.) benzetimin çalışması sırasında her cihazın aktivitesine göre bir çıktı elde edilmektedir. PowerTOSSIM her cihaz tarafından çalıştırılan CPU çevriminin tahmini için yeni bir kod dönüşüm tekniği kullanmaktadır ve algılayıcı düğümlere benzetim imkânı sunarak bu pahalı komut-seviye işlemini elimine etmektedir. PowerTOSSIM MICA2 algılayıcı düğüm platformu üzerinde donanım enerji tüketiminin detaylı bir modelini içermektedir [3].

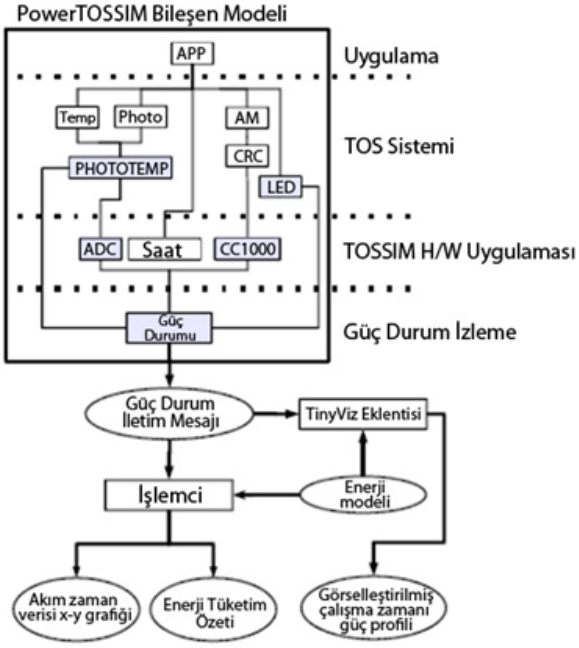
PowerTOSSIM yeni bir kod dönüşüm tekniği kullanarak her düğüm için CPU çevrim çalışmasını tahmin etmektedir. Son olarak, her düğümün aktivitesini detaylı olarak incelemek için donanım enerji tüketimine göre detaylı bir model sunmakta ve her düğümün enerji tüketimi konusunda verimli sonuç alınması mümkün olmaktadır[4]. Bu enerji modeli diğer donanım platformları içinde kolayca değiştirilebilir olma özelliği sunmaktadır.

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

PowerTOSSIM'de TinyOS işletim sistemi ve TOSSIM benzetim ortamı ele alınmıştır. TinyOS bir benzetim ortamı olan TOSSIM'i desteklemektedir. TOSSIM'de TinyOS uygulamaları bir benzetim cihazı üzerinde olaya dayalı benzetim ile doğrudan derlenmektedir. Bu tasarım ile TinyOS'un bileşen odaklı özelliğinden faydalanılmaktadır. Yani TinyOS bileşenlerinin donanıma kolay erişiminden faydalanılmaktadır. TOSSIM donanım bileşenlerini örneğin basit radyo yığını, algılayıcılar ve diğer çevre birimlerinin benzetimini desteklemektedir. PowerTOSSIM ise TOSSIM ve TinyOS bileşen modellerini enerji tüketimini takip etmek için kullanmaktadır[4-5].

Var olan benzetim ortamlarına örnek olarak ns2, TOSSIM ve Atemu verilebilir. Bu ortamlar sayesinde algılayıcı ağların davranışları detaylı olarak anlaşılabilir. Çünkü bu ortamlar çeşitli derecelerde ölçeklenebilirliği ve gerçekçiliği sağlamaktadırlar.

PowerTOSSIM Mimarisi:



Şekil 1. PowerTOSSIM Mimarisi

Şekil 1'de PowerTOSSIM mimarisi gösterilmektedir. PowerTOSSIM, benzetimi yapılan düğümlerin donanım bileşenlerinin güç durumunu takip etmekte ve bu işlemi de belirli güç durum dönüşümü mesajı oluşturarak yapmakta ve bu mesajlar benzetim sırasında kayıt altına alınmaktadır. Bunda başarı ise TOSSIM'in donanım bileşenlerini yeni bir bileşen çağırarak benzetim yapmasıyla ortaya çıkmaktadır. Bu yeni bileşen Güç-Durumudur. Bu bileşen her düğüm için donanım güç durumunu izlemekte ve çalışma sırasında bunları bir dosyaya kaydetmektedir. CPU kullanımının tahmini bununla daha çok ilişkilidir. CPU profili benzetim tarafından çalıştırılan temel bloklar sayesinde ilgili düğümün çevrim sayısının bulunması ile başarılmaktadır[5].

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

2.1. Enerji Tüketiminin Ana Kaynakları

Radyo, düğümlerdeki önemli güç tüketicisidir ve radyonun durumunu doğru şekilde takip etmek algılayıcı ağlarda herhangi bir güç görüntü aracının başarılı olması için gereklidir. MICA2 platformu farklı bir radyo cipi kullanmaktadır. CC1000 radyo 3 temel durum içermektedir. Bunlar iletim, dinleme ve uyumadır. Uygulamaya göre değişmekle birlikte uyuma durumunda sabit enerji tüketimi olmakta iken dinleme ve iletim durumlarında enerji tüketimi artmaktadır [6].

CPU güç durumu için MICA2 tarafından kullanılan Atmega 128LCPU'da 7 tane farklı güç durumu bulunmaktadır. Bunlar; aktif, IDLE, ADC NOISE REDUCTION, POWER_DOWN, POWER_SAVE, STANDBY ve EXTENDED STANDBY'dır. Bir kaç TinyOS uygulaması bu durumlardan faydalanırken PowerTOSSIM'de bunların yakalanması daha anlaşılabilir olmaktadır. Varsayılan CPU aktif durumdayken yani komutları çalıştırırken ve IDLE durumdayken bir kesme gelmesini bekleme durumunda olabilmektedir. Genelde CPU çevrim tahminine bakılarak aktif durumda ne kadar zaman kaldığı belirlenebilmektedir[7]. Diğer durumda ise CPU IDLE durumunda olmaktadır. Diğer güç durumları belirli CPU kayıt ayarları tarafından kayıt altına alınmış olur. Burada iki standart vardır ve TinyOS bileşenleri bunlardan yani HPLPowerManagament ve Snooze bileşenlerinden yararlanmaktadır. Güç durumu için bu iki bileşenden birini çağırmak gerekmektedir. Sebebi ise CPU'nun düşük güç durumuna giriş ve çıkışını yakalamak hesaplama açısından önemlidir.

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

EEPROM güç durumu için MICA2 ek olarak bir EEPROM içermekte bunu da veri kaydetme için yapmaktadır. TinyOS EEPROM modülü sayesinde düğümün kendi EEPROM'una yazma ve okumada ne kadar süre harcadığını takip edebilir. EEPROM'a yazma ve okuma farklı miktarda güç tüketmekte ve farklı zamanlarda bitmektedir. Bundan dolayı bu durumların her dönüşümü güç durumu tarafından kaydedilmektedir [8].
ADC güç durumu için analog-dijital çevrimi fiziksel çevreden örnek anolog algılayıcı veri almak için kullanılmaktadır. MICA2'de ADC CPU'ya yerleşiktir ve aktifken de ayırt edilebilir miktarda güç tüketmektedir. Alternatif algılayıcı düğüm platformları harici bir ADC'yi kapsayabilmekte ve bu ADC'ler bağımsız olarak aktif veya kapalı olabilmektedirler. Bundan dolayı TinyOS ADCC bileşenini ADC aktifken ki zamanı takip etmek için düzenlemiştir[9].

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

Orijinal TOSSIM kodunda bir kaç değişiklik ile bu çeşitli cihazların güç durumları takip edilebilmektedir.

2.2. Uyku Durumları

MCU (Micro Controller Unit-Mikro Denetleyici Birimi) içerisinde 6 tane güç kipi durumu bulunmaktadır. Bu uyku kipleri, MCU kontrol yazmaçlarındaki (SM2,SM1,SM0) 3 bitin değerine bağlı olarak değişmektedir. Bu üç bit uyku yazmacı olarak çalışmaktadır. SE bitinin 1 olması CPU'nun uyku kipine girmesi demektir (Bu da uyku komutlarının çalıştırılması ile mümkün olmaktadır). Tavsiye edilen ise SE bitinin uyku komutları çalıştırılmadan ayarlanması yani 1 yapılması ve uyanma işleminden sonra temizlenmesidir [9-10].

IDLE Kipi: Ne zamanki uyku yazmacının değeri 000 ise ve uyku komutları çalıştırılırsa işlemci IDLE durumuna geçer. CPU durur ancak SPI, USART, Analog karşılaştırıcı, İki telli kanal seri arayüzü, Zamanlayıcı/Sayaçlar, Gözetleme ve kesme sistemleri işlemlerine devam ederler. Bu kip clkCPU ve clkflash'ı durdurur. Eğer ki analog karşılaştırıcı kesmesi uyanma için gerekli değilse, analog karşılaştırıcılarda bulunan ACD bitiyle, durum kontrol yazmaçları(ACSR) ayarlanarak kapatılabilir. Uyanma işlemi ise harici kesmelerle, yani örneğin zaman aşımı ve USART iletişim tamamlama ile gerçekleşir[9].

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

seviye kesmesi veya INT3:0 harici kesmesi MCU'yu bu uyku kipinden uyandırır[10].

POWER_DOWN Kipi: Ne zamanki uyku yazmaçlarının değeri 010 ise ve uyku komutları çalıştırılırsa CPU bu kipe geçer. Bu kipte, harici Oscillator durdurulur ancak harici kesmeler, İki telli kanal seri arayüz adresleme ve gözcü sıfırlama aktif ise işlemine devam eder. Bu uyku kipi bütün oluşturulan saatleri durdurur. Sadece asenkron modüllerin işlemine izin verir. CPU'nun uyanması için bir kesme geldiği zaman uyanma işlemi için değişen seviyenin düzenlenmesi gerekmektedir. Bu da belirli bir zaman aralığında olmaktadır. Uyanma işlemi ise harici bir reset, Gözcü sıfırlama, Voltaj-Düşüklüğü sıfırlama, İki telli kanal seri arayüz adresleme kesmesi, INT7:4 üzerinde harici seviye kesmesi veya 3:0 da harici kesme durumunda gerçekleşir[10-11].

POWER_SAVE Kipi: Ne zamanki uyku yazmaçlarının değeri 011 ise ve uyku komutları çalıştırılırsa, CPU bu kipe girer. Bu kip POWER_DOWN kipiyle bir şey hariç aynıdır. Eğer Zamanlayıcı/Sayaç0 saati asenkron ise ASSR'deki AS0 biti 1 değeri şeklinde ayarlanır ve Zamanlayıcı/Sayaç0 uyku durumunda da çalışır. Uyanma işlemi ise harici bir sıfırlama yani Zamanlayıcı/Sayac0 kesmesi ile olmaktadır[12].

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

EXTENDED STANDBY Kipi: Ne zamanki uyku yazmaçlarının değeri 111 ise ve uyku komutları çalıştırılırsa CPU bu kipe geçer. Bu kip POWER_SAVE kipi ile aynıdır bir şey hariç o da oscillator burada çalışmaya devam eder. İşlemcinin uyanması bu kipte 6 saat çevriminden sonra olmaktadır.

Enerji tüketimini azaltmak için aşağıdaki işlemlerin değerlendirilerek kullanılması gerekmektedir.

ADC: Eğer açıksa tüm kiplerde açıktır. Gücü koruyabilmek için uyku kipine geçilmeden ADC kapatılmalıdır.

Analog Karşılaştırıcı: IDLE veya ADC NOISE REDUCTION kipe geçerken, Analog karşılaştırıcı eğer kullanılmıyorsa kapatılmalıdır. Diğer uyku kiplerinde Analog karşılaştırıcı zaten otomatik kapatılmaktadır. Fakat Analog Karşılaştırıcısı İç Gerilim İlişkisi için girdi olarak kullanılmaktadır. Bu yüzden bütün uyku kiplerinde kapatılmalıdır.

Voltaj-Düşüklüğü Algılayıcısı: Eğer ihtiyaç yoksa bu modül kapatılmalıdır. Eğer ki Voltaj-Düşüklüğü Algılayıcısı BODEN FUSE tarafından kullanılıyorsa, tüm uyku kiplerinde açık olmalıdır.

İç Gerilim İlişkisi: İç gerilim ilişkisi Voltaj-Düşüklüğü Algılayıcısı, Analog Karşılaştırıcı veya ADC tarafından ihtiyaç duyulursa aktif olur. Eğer bu modüller kapalıysa bu modül kapatılmalı ve enerji tüketmemelidir.
Gözcü Saati: İhtiyaç yoksa bu modül kapatılmalıdır. Eğer açıksa tüm uyku kipleri için açık olmaktadır.

Port Pins: Uyku kiplerine geçilirken bütün port pinleri minimum güç tüketecek şekilde ayarlanmalıdır. Tüm uyku kiplerinde I/O saati ve ADC saati durdurulursa, cihazın girişleri kapalı olmalıdır. Buda bize ihtiyaç yokken giriş tarafından enerji tüketilmediğini garantiler. Bazı durumlarda giriş için uyanma şartlarını yakalamak gerekmektedir bu durumda açık olabilmektedir [13-14].

2.3.TinyOS İşletim Sisteminde Güç Yönetim İşlevselliği

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

- Radyonun kapandığından (CC1000RadioC. StdControl. stop() komutunun çağrılması gerekmektedir.), Bütün yüksek hızlı kesmelerin kapatıldığından, SPI kesmesinin kapalı olduğundan ve Task kuyruğunun boş olduğundan emin olunması gerekmektedir.

- Bunlardan sonra düğüm bir sonraki saat darbesine kadar uyku kipine geçer. Radyoya ek olarak her hizmette kapatılır. Stop() fonksiyonu içinde, her hizmetin HPLPower Management.adjustPower() komutunu çağırması gerekmektedir. Uygulamaların HPLPower Management.adjustPower() komutunu çağırmaya ihtiyacı yoktur [4-15].

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

Tablo1. MICA2'nin enerji modeli

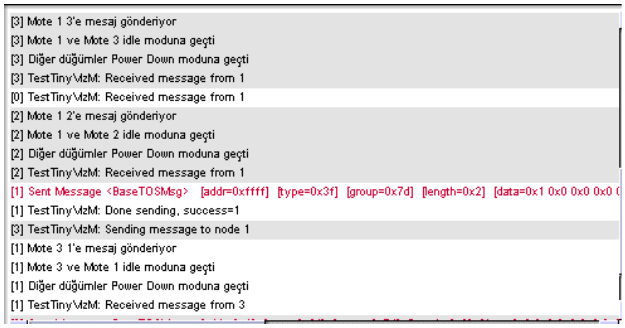
Enerji Kipleri	**joule**
CPU_ACTIVE	8.93
CPU_IDLE	4.13
CPU_ADC_NOISE_REDUCTION	1.0
CPU_POWER_DOWN	0.103
CPU_POWER_SAVE	0.110
CPU_STANDBY	0.216
CPU_EXTENDED_STANDBY	0.223

Tablo1'de MICA2'nin enerji modeli gösterilmektedir. Güç Kipleri için uygulanan standart değerleri göstermektedir.

3. UYKU KİPLERİNE GÖRE ENERJİ TÜKETİMİ BENZETİMİ

Burada MICA2 için enerji tüketimi IDLE ve POWER_DOWN kipleri için gerçekleştirilmiş ve düğümün uyuma zamanına göre enerji tüketimleri gösterilmiştir. Test uygulamasının ekranda

gösterilebilmesi için “make pc” komutu çalıştırılmış ve herhangi bir hata alınmayınca güç tüketimlerini elde etmek için “export DBG=power” komutu çalıştırılmıştır. Daha sonra “build/pc/main.exe –gui –p 4” ile 4 tane düğümün ekranda birbirleri ile bağlanma işlemi, Gui (Graphical User Interface) kullanılarak yapılmış ve güç tüketimlerini elde etmek için ise –p komutu eklenmiştir. Alttaki ekranda TinyViz ekranının çalışması için “java net.tinyos.sim.TinyViz” komutu çalıştırılmıştır[9-11].



Şekil 2.Test Uygulamasının Power State kullanımı ve güç kipleri arasında geçiş

Şekil 2’de test uygulamamız olan TestTinyViz uygulamasının debug mesaj kısmı gösterilmektedir. Daha açıklayıcı olması açısından hangi düğümün hangi düğüme mesaj attığı gösterilmiştir.

Dolayısıyla 2 düğüm iletişim yaparken diğer düğümlerin hangi kipe geçtiği belirlenmiş ve ekrana çıktı olarak verilmiştir. Burada bazı uygulamalar POWER_DOWN bazıları da POWER_SAVE kipini kullanmaktadır. Genel literatür araştırmalarından faydalanılarak en fazla enerji korunumunun POWER_DOWN kipte elde edildiği görülmüştür.

IDLE kipte enerji tüketimi en düşük olarak elde edilmiştir. Burada IDLE güç kipinde CPU ve belirli saatler durdurulmaktadır. Burada, 10 tane mesaj gönderimi sırasında tüketilen enerji miktarları gösterilmektedir. Genel olarak çıkan sonuçlarda devamlı gönderen düğümün enerji seviyesi hep aynı çıkmaktadır. Ancak düğümün mesaj oluştururken bazen fazla zaman geçebilmekte ve ayrıca alıcı düğümün mesajı alırken geçirdiği zaman içerisinde alıcı düğümün güç kiplerindeki enerji tüketimi farklı çıkmaktadır. Burada bir kaç senaryo üzerinde işlemler gerçekleştirilmiştir.

moteid	radio	cpu	led	eeprom	total
0	82,71	33,24	0,00	0,00	115,94
1	107,51	43,15	0,00	0,00	150,65
2	25,57	10,26	0,00	0,00	35,83
3	112,41	45,12	0,00	0,00	157,53

Şekil 3. IDLE kip durumunda çoklu işlemde enerji tüketimi

Şekil 3’te ortalama 10 saniyede rastgele yapılan iletişimde tüketilen toplam enerji miktarları gösterilmektedir. Şekilde görüldüğü gibi en fazla enerji tüketimi 1. ve 3. düğümde olmaktadır.

moteid	radio	cpu	led	eeprom	total
0	22,04	8,85	0,00	0,00	30,89
1	88,97	35,70	0,00	0,00	124,67
2	87,17	34,99	0,00	0,00	122,15
3	27,70	11,12	0,00	0,00	38,82

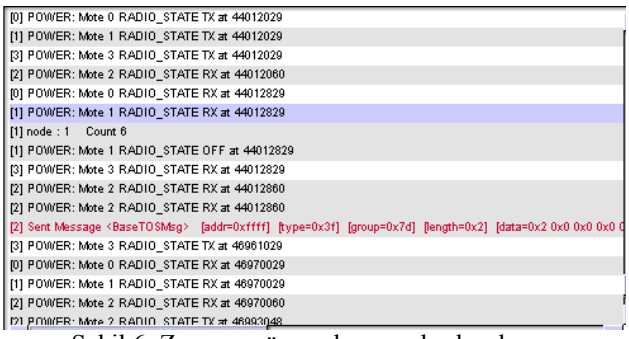
Şekil 4. POWER_DOWN kip durumunda çoklu iletişimde enerji tüketimi

Şekil 4’te 4 tane düğümün rastgele iletişimi esnasında enerji tüketimleri gösterilmektedir. En fazla alımı 1. ve 2. Düğümler yapmasına rağmen ortalama 10 saniyelik bir iletişimde genel anlamda Şekil 3’te gösterilen ortalama 10 saniyelik iletişimden daha iyi sonuç verdiği ve daha az enerji tükettiği görülmektedir.

Tablo 2. Güç kiplerinde enerji tüketimi

Kipler	mote id	radio	cpu	led	toplam
POWER_DOWN	0	64,27	25,83	0,0	90,10
	1	126,96	50,96	0,0	177,92
IDLE	0	141,98	57,07	0,0	199,05
	1	126,96	50,96	0,0	177,92

Tablo 2’de güç kiplerinden 2 tanesi IDLE ve POWER_DOWN kipleri kullanılmış ve tüm güç kipleri kapatılarak iki düğüm arasındaki iletişimde enerji tüketimleri gösterilmiştir. Tablo 2’de gösterildiği gibi alıcı gönderen düğümün enerji tüketimleri sabitlendiğinde diğer düğümleri güç kiplerine göre 10 mesaj gönderiminde toplam tüketilen enerji miktarları gösterilmektedir. Buradaki durumda paketler farklı üretildiği için farklı zaman aralıkları çıkmakta ve bu durumda üretilen düğümlerin enerji tüketimleri gösterilmektedir. Yani paket üretim süresinin farklı olması da aynı zamanda enerji tüketimini etkilemektedir. Burada Tablo 2’de görüldüğü gibi tüketilen enerji miktarı kullanılan güç kipine göre değişmektedir.



Şekil 6. Zamana göre radyonun durdurulması

Şekil 6’da gösterildiği gibi düğüm 1 saat çevrimi 6 olduğunda radyosu durdurulmaktadır. Ve benzetim süresincede açılmamaktadır.

moteid	radio	cpu	led	eeprom	total
0	225,91	90,82	0,00	0,00	316,73
1	12,67	94,08	0,00	0,00	106,75
2	202,70	81,49	0,00	0,00	284,18
3	180,91	72,73	0,00	0,00	253,63

Şekil 7. Radyonun durdurulması zamanında enerji tüketimi

Şekil 7'de gösterildiği gibi düğüm 1'in değeri hiç değişmemektedir. Diğerlerine göre çok az bir enerji tüketimi vardır. Burada "PowerState.radioStop()" komutu kullanılmıştır ve sadece düğüm1 için bu komut kullanılmıştır. 12,67'lik enerji harcaması ise düğümün ilk açılması esnasında ortaya çıkmaktadır. Burada eeprom ve led durumlarında da enerji tüketilirken benzer komutlarla durdurmak ve enerji tüketimini azaltmak mümkün olmaktadır. CPU ise POWER_DOWN kipi kullanılarak enerji tüketimi en alt seviyeye indirilebilmektedir.

4. SONUÇLAR

Kablosuz algılayıcı ağların topolojilerinin oluşturulması genelde rastgele olmaktadır. Milyonlarca düğümün kolay ulaşılamayacak yerlere atılması sonucu ortaya çıkan en önemli problem bu düğümlerin pillerinin şarj edilememesi ve değiştirilememesidir. Düğümlerin ömrünün tükenmesi ağın ömrünü doğrudan etkilemektedir. Gecikme gibi kablosuz ağlarda önemli olan konular algılayıcı ağlarda geri plana düşmektedir.

Bu çalışmada TinyOS işletim sisteminde bulunan güç modelleri incelenmiş ve düğümlerin zamanlayıcıya göre uyku kipine geçmeleri ve ayrıca durdurulup tekrar açılmaları gerçekleştirilmiştir. Uyku kipine geçiş sayesinde en önemli enerji tüketimine sebep olan istem dışı alım büyük ölçüde önlenebileceği görülmüştür. Ayrıca bu çalışmada kullanılan güç kipleri MICA2 düğümler için kullanılmaktadır.

KAYNAKLAR

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

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

[3] S. Changsu, K. Young-Bae, "A traffic aware, energy efficient MAC protocol for wireless sensor Networks", **IEEE International Symposium on Circuits and Systems(ISCAS)**, Vol. 3,2975 – 2978, 2005

[4] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, (2002), "Wireless Sensor Networks: A Survey", Computer Networks, Vol. 38, 393-422, 2002

[5] A. Karagiannis, S. Kokkorikos, P. Constantinou, "Energy Consumption Analysis and Optimization techniques for Wireless Sensor Networks", **ISCC: 2009 IEEE Symposıum On Computers and Communıcatıons**, Vol. 1 AND 2, 8-14, 2009

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

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

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

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

[10] O. Landsiedel, K.Wehrle, S. Götz, "Accurate Prediction of Power Consumption in Sensor Networks", Proceeding EmNets '05 Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors, 37-44, 2005

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

[12] B. Zebbane, M. Chenait, N.Badache, H. Zeghilet, "Topology Control Protocol for Conserving Energy in Wireless Sensor Networks", **ISCC: 2009 IEEE Symposıum On Computers And Communıcatıons**, Vol.1 AND 2, 716-719, 2009

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

[14] G. J. Pottie, W. J. Kaiser, "Wireless Integrated Network Sensors", Commun. ACM, Vol. 43, 551 – 558, 2000

[15] L.H.A. Correia, D.F. Macedo, A.L. dos Santos, A.A.F. Loureiro, J.M.S. Nogueira, "Transmission power control techniques for wireless sensor networks", Computer Networks, Vol. 51(17) , 4765-4779, 2007.