ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

SYSTEM DESIGN FOR INTERNET OF THINGS AND NETWORK CODING APPLICATIONS IN THE WIRELESS PERSONAL AREA NETWORKS

M.Sc. THESIS

Görkem ÖZVURAL

Department of Electronics and Communications Engineering

Telecommunications Engineering Programme

MAY 2015

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

SYSTEM DESIGN FOR INTERNET OF THINGS AND NETWORK CODING APPLICATIONS IN THE WIRELESS PERSONAL AREA NETWORKS

M.Sc. THESIS

Görkem ÖZVURAL
(504111316)

Department of Electronics and Communications Engineering

Telecommunications Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Güneş KARABULUT KURT

MAY 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

NESNELERİN İNTERNETİ İÇİN SİSTEM TASARIMI VE KABLOSUZ KİŞİSEL ALAN AĞLARINDA AĞ KODLAMA UYGULAMALARI

YÜKSEK LİSANS TEZİ

Görkem ÖZVURAL
(504111316)

Elektronik ve Haberleşme Mühendisliği Ana Bilim Dalı

Telekomünikasyon Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Güneş KARABULUT KURT

MAYIS 2015

Görkem ÖZVURAL, an M.Sc. student of ITU Graduate School of Science Engineering and Technology 504111316 successfully defended the thesis entitled **"SYSTEM DESIGN FOR INTERNET OF THINGS AND NETWORK CODING APPLICATIONS IN THE WIRELESS PERSONAL AREA NETWORKS"**, which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Güneş KARABULUT KURT** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. İbrahim ALTUNBAŞ** Istanbul Technical University	.....
	Prof. Dr. Emin ANARIM Boğaziçi University	.....

Date of Submission : **4 May 2015**
Date of Defense : **22 May 2015**

To my family and wife,

FOREWORD

This thesis is written as completion to the Master of Science in Telecommunications Engineering Programme at Istanbul Technical University and the thesis is supported in part by TUBITAK under Grant 113E294 and Vestel Electronics Inc. The master program focuses on the basis of both wired and wireless telecommunication systems not only in theoretical but also in practical manner. The subject of this thesis falls within the scope of Internet of Things network design for more efficient communication between devices and thesis study compromises implementation of the proposed system. I have chosen to study more practicable project to be able to transfer my experience gained through work in industry and redound an IoT development environment for researchers studying in Wireless Communication Research Laboratory (WCRL/THAL) at Istanbul Technical University.

I would like to thank my supervisor Assoc. Prof. Dr. Güneş Karabulut Kurt for giving me valuable advices and support during my studies. Also I always feel the support of my family and my wife, I owe a debt of gratitude to them.

May 2015 Görkem ÖZVURAL

TABLE OF CONTENTS

Page

ABBREVIATIONS

AP : Access Point
API : Application Programming Interface
BB : Battery Board
CPU : Central Processing Unit
CSMA-CA : Carrier Sense Multiple Access with Collision Avoidance
EB : Evaluation Board
EM : Evaluation Module
DMA : Direct Memory Access
DMIPS : Dhrystone MIPS
FSM : Finite-state Machine
GPIO : General Purpose Input/Output
HetNet : Heterogeneous Network
ISM : Industrial, Scientific and Medical
JTAG : Joint Test Action Group
kbps : Kilobit per second
LAN : Local Area Network
LLC : Logical Link Control
LR-WPAN : Low Rate Wireless Personal Area Network
MAC : Medium Access Control
Mbps : Megabit per second
MCU : Microcontroller Unit
MIPS : Million instructions per second
MSB : Most Significant Bit
OSI : Open Systems Interconnection
PCB : Printed Circuit Board
PHY : Physical Layer
PNC : Physical-layer Network Coding
PSDU : PHY Service Data Unit
RAM : Random Access Memory
RF : Radio Frequency
ROM : Read Only Memory
RLNC : Random Linear Network Coding
RTOS : Real Time Operating System
SiP : System in Package
SMD : Surface Mount Device
SOA : Service Oriented Architecture

SoC : System on Chip
TDMA : Time Division Multiple Access
THAL : Telsiz Haberleşme Araştırma Laboratuvarı (WCRL in English)
TI : Texas Instruments
uC : Microcontroller
WAN : Wide Area Network
WCRL : Wireless Communication Research Laboratory
WICED : Wireless Internet Connectivity for Embedded Devices
WiFi : Wireless Fidelity
WLAN : Wireless Local Area Network
XOR : Exclusive OR
ZNP : ZigBee Network Processor

LIST OF TABLES

Page

LIST OF FIGURES

Page

SYSTEM DESIGN FOR INTERNET OF THINGS AND NETWORK CODING APPLICATIONS IN THE WIRELESS PERSONAL AREA NETWORKS

SUMMARY

Internet of Things (IoT) is a promising concept to meet the needs of a smarter world in near future. Through the way from development to the deployment of IoT devices, wireless technologies seem to be dominant over wired infrastructures. Plug and play nature of wireless communication makes it more preferable over other alternatives. Almost every IoT device will have at least one radio interface and they will be condensed to the same spatial location. Not only the number of wireless IoT devices increase, but also people move in more crowded multistorey buildings which causes concentration of wireless nodes in same vicinity. Therefore, efficient usage of the radio spectrum will become more important than ever for upcoming decades.

As we can easily deduce, efficient usage of the radio spectrum is not only about physical layer protocols but also upper layers that generate payload for the physical layer protocols. For this reason, cross layer approaches gain importance because efficiency is generally provided by multilayer analysis and improvements in telecommunication systems. As a high capacity alternative to conventional routing techniques, network coding provides more efficient usage of radio spectrum for multicast communication between nodes.

The main idea behind network coding is processing the received packets in the intermediate nodes instead of simply storing and forwarding within the scope of network layer. With the use of coding techniques, number of transmissions can be reduces for same amount of data transferred through the network. This provides not only efficient usage of radio spectrum for same amount of data but also increases data throughput in unit of time.

Within the scope of this study, network coding is applied in a heterogeneous network formed by hybrid radio nodes. Low capability hardware is preferred instead of sophisticated radio nodes, to be able to adapt proposed system to the field easily.

NESNELERİN İNTERNETİ İÇİN SİSTEM TASARIMI VE KABLOSUZ KİŞİSEL ALAN AĞLARINDA AĞ KODLAMA UYGULAMALARI

ÖZET

Geçtiğimiz yüzyılın en önemli yeniliği olan İnternet ve bilgisayarlar arası ağ altyapısı, dünya tarihi boyunca yaşanan gelişmeler arasında değerlendirildiğinde, kısa bir süreçte bu denli geniş kapsamlı değişime sebep olan nadir teknolojik gelişimler arasındadır. Bilgisayarlar arası iletişime imkan tanıyan cihazların ticari uygulamalarda kullanılabilecek duruma gelmesi, farklı ihtiyaç ve uygulamalara cevap verebilecek haberleşme protokollerinin geliştirilmesinin önünü açmıştır. Bunun sonucunda, internet kullanımının yaygınlaşması ve birçok farklı sektörü etkisi altına alması kaçınılmaz olmuştur.

İnter-Net, dünya üzerindeki bilgisayarların elektronik bir altyapı ile birbirlerine bağlanarak bilgi alışverişi yapmalarını sağlayan bir iletişim ağıdır. Internet'in ortaya çıkışı, Amerikan Savunma Bakanlığı'nın 1969'da bilgisayar ve askeri araştırma projelerini desteklemek için ARPANET adında paket anahtarlamalı bir ağ tasarımı projesini gerçekleştirmesine dayanmaktadır. Bu ağ, üniversiteler ve araştırma kuruluşlarının bilgisayarlarıyla birlikte genişleyerek büyümüştür.

Internet, başlangıcından beri birçok belirgin aşama geçirmiştir. Bunlar başlıca şu ana başlıklar halinde sıralanabilir:

1. Aşama: Araştırma dönemi; Advanced Research Projects Agency Network (ARPANET) zamanları. Günümüzde halen kullanılmakta olan TCP/IP protokol kümesinin temelleri 1973 yılında Stanford Üniversitesi'nde başlatılan Internetworking projesiyle atılmıştır. 1973-1978 yılları arasında IPv1, IPv2 ve IPv3 versiyonları geliştirilmiştir. Stabilite sorunlarının da giderilmesiyle 1980 yılında IPv4 protokol kümesi ortaya çıkmıştır.

2. Aşama: Şirketlerin, firmaların kendileri hakkında bilgilendirme yaptığı dönem olarak adlandırılabilir. Herkesin bir alan adı alıp, ne yaptığını, nasıl yaptığını anlatarak daha geniş kitlelere tek yönlü olarak bilgi aktarma çabasında oldukları bir zaman dilimi olarak tanımlanabilir.

3. Aşama: Bu dönem de, web artık giderek statik ve durağan olmaktan çok interaktif olmaya doğru yol almakta, sunulan ürün ve servislerin satın alınma işlemlerinin gerçek zamanlı gerçekleştiği altyapı olanaklarının yaygınlaşarak geniş kitlelere sunumu sağlanmaktadır. İşte tam bu zaman dilimi içerisinde eBay ve Amazon.com benzeri uygulamalar çoğalmışlardır.

4. Aşama: "Sosyal Medya, Semantik Web" kavramları yaygınlık kazanmaktadır. Kullanıcıların birbirleri ile metin, foto, video gibi medyaların paylaşımını sağlayan YouTube, Facebook, Twitter gibi şirketlerin giderek hem popüler olması hem de para kazanır duruma gelmeleri bu dönemde var olmuştur.

5. Aşama: Bundan önceki aşamalarda haberleşme teknolojilerinin gelişmesi ve yaygınlaşmasıyla, "her zaman" ve "her yerde" bağlanabilirlik kavramları ortaya çıkmıştır. Nesnelerin internetiyle birlikte bu kavramlara ek olarak "her şey" kavramı da önem kazanmış ve akla gelebilecek tüm nesnelerin internete bağlanması fikri ortaya atılmıştır.

İnternetin zaman içerisinde evrimleşmesini takiben, "Nesnelerin İnterneti" (Internet of Things) kavramı ilk kez 1999 yılında Kevin Ashton tarafından P&G şirketi için hazırlanan bir sunumda kullanılmıştır. Bu sunumda şirketin tedarik zincirinde RFID teknolojisi uygulamasının firmaya faydaları sıralanmakta ve kullanımı önerilmekteydi. Nesnelerin İnterneti, günlük yaşantımızda karşılaştığımız problemleri, yazılım uygulamalarının, günlük objelerimizi ve internetin bağlanabilirliğini kullanarak çözdüğü devrimsel bir teknoloji olarak tanımlanabilir. Yani nesnelerin interneti, hayatımızı daha güvenli, akıllı ve üretken hale getirerek yaşama ve çalışma biçimimizi değiştirecek bir teknolojidir. Nesnelerin İnterneti, benzersiz bir şekilde adreslenebilir şeylerin/nesnelerin kendi aralarında oluşturduğu, dünya çapında yaygın bir ağ ve bu ağdaki nesnelerin belirli bir protokol ile birbirleriyle iletişim içinde olmalarını sağlar.

Nesnelerin İnterneti, üç ana bileşenden oluşur:

- Nesneler
- Nesneleri birbirine bağlayan iletişim ağları
- Nesnelerden nesnelere akan verileri kullanan bilgisayar sistemleri

Yapılan araştırmalara göre bugün İnternete 25 milyar cihazın bağlı olduğu tahmin edilmekte ve bu rakamın, 2020 yılına gelindiğinde, 50 milyar cihaz seviyesine çıkması öngörülmektedir. Aynı araştırmalara göre; 2003 yılında dünyada kişi başına düşen, birbirleriyle bağlantılı cihazların oranı 0,08 iken bu oranın 2020 yılında 6,48 olması beklenmektedir. Ayrıca 2020 yılında, 20 adet tipik ev cihazının üreteceği bilgi trafiğinin, 2008 yılında üretilen tüm internet trafiğinden daha fazla olacağı tahmin edilmektedir. Bu veriler ışığında, bağlı cihaz uygulamaları arasında oluşacak veri trafiğinin daha verimli, daha güvenilir ve veri bütünlüğünü bozacak herhangi bir güvenlik açığı oluşmadan transferi gelecekte daha büyük önem arzedecektir.

Yenilikçi yaklaşımların nesnelerin internetinin geleceğini belirleyeceği aşikardır. Bu tez çalışmasıyla, nesnelerin interneti konsepti için farklı disiplinlerdeki ve uygulamalardaki yenilikçi yaklaşımlar harmanlanarak daha verimli bir ekosistem oluşturulmasına katkı sağlanması amaçlanmıştır. Proje kapsamında, İTÜ Telsiz Haberleşme Araştırma Laboratuvarı bünyesine bulut uygulamaları ve servisleri, mobil arayüz Android uygulamaları, sensör ve aktüator cihazları da dahil olmak üzere nesnelerin interneti alanında çalışmalara ön ayak oluşturabilecek küçük bir ekosistem kazandırılmıştır. Bu ekosistem içerisindeki çalışmaları özetlemek gerekirse:

Sunucu tarafında; genişletilebilir uygulama ve katma değerli servis entegrasyonuna olanak sağlayan bir altyapı hazırlanmıştır. Nesnelerin uzaktan yönetimi ve nesneler tarafından oluşturulan büyük verinin kaydedilmesi ve işlenmesi için SQL tabanlı bir veritabanı altyapısı sağlanmış, gerekli bulut servisleri ile entegre edilmiştir. Nesneler tarafından oluşturulacak büyük veri üzerinde veri madenciliği yapılması ve bu verilerin katma değerli servisler olarak sunulması aşamasında gerekecek altyapı

oluşturulmuştur. Kurulan sunucu ve bulut servisler altyapısı sanal makine üzerinde çalıştırılarak, sistemin taşınabilirliği garanti edilmiştir. Böylece sunucu ve bulut servis altyapısı, makine bağımsız şekilde istenen herhangi bir hosting servis sağlayıcısıyla çalıştırılabilir kılınmıştır. Bu yapı ileriye dönük geliştirmelerde ve sistemin geniş alan ağları üzerinden test edilmek istenmesi durumunda entegrasyon açısından kolaylık sağlayacaktır.

Sensör ağ kısmında; IEEE 802.15.4 standardı üzerinde çalışan ZigBee protokolü kullanan sensör ve aktüatör cihazları için, gömülü sistem tabanlı WiFi - ZigBee IP Ağ Geçidi düğümü tasarlanıp gerçeklenmiştir. Bu ağ geçidi düğümü ile sensör ve aktüatör nesneleri birbirine bağlayan kablosuz kişisel alan ağında yepyeni bir yaklaşım olan ağ kodlama tekniği ile ilgili hem teorik hem de deneysel çalışmalar gerçekleştirilmiştir. Deneysel çalışmalar kapsamında, 1 ağ geçidi düğümü, 1 sensör düğümü ve 1 aktüatör düğümü ile oluşturulan 3 düğümlü ağ üzerinde ağ kodlama tekniklerinden biri olan XOR kodlama tekniği denenmiş, sensörler arası veri trafiğinde daha az iletim döngüsü ve daha düşük güç tüketimiyle aktarım sağlanmıştır. Uygulanan bu senaryo, akıllı evlerde sensörler ve aktüatörler arası veya sensör cihazlarla data toplayıcı ağ geçidi cihazları arası veri paylaşım senaryosu için oldukça verimli bir data transferi alternatifi sunmaktadır.

Teorik çalışmalar kapsamında MATLAB ortamında, kablosuz kişisel alan ağlarında rastgele ağ kodlama tekniğinin test edilebilmesine imkan sağlayan simulasyon ortamı geliştirmesi yapılmıştır. Bu geliştirmeler yapılırken farklı ağ topolojilerine ve düğüm sayılarına destek verebilmek amacıyla dinamik bir kodlama sağlanmıştır. Rastgele ağ kodlama tekniğinin başarımını test etmek için temel olarak kanalın sembol silme olasılığı ile sembol hata oranının kıyaslanması sağlanmıştır. Farklı Galois alanı boyutları için sembol hata oranı kıyaslamalı olarak test edilerek rastgele ağ kodlama başarımı doğrulanmıştır.

Son olarak, ağ geçidi düğümü tasarlanırken özellikle piyasada rahatlıkla bulunabilen WiFi ve ZigBee modülleri tercih edilmiş, çalışma sonucunda ortaya çıkabilecek faydalı kullanım senaryoları ve tekniklerin kolayca kullanıma geçirilmesi ve adapte edilebilmesi hedeflenmiştir. Böylece, akademik bir çalışmanın endüstriyel alanda da yer bularak kullanıma geçirilmesi sürecine katkı sağlanmıştır.

Sensörler ve aktüatörler kısmında; sıcaklık sensörü düğümü ve çok amaçlı elektriksel aç-kapa düğümü donanım ve yazılımları gerçekleştirilmiş, bu cihazlar sensör ağına dahil edilmiştir. Ayrıca XOR kodlama tekniği ile bu iki düğüm arası veri transferi gerçekleştirilerek, ev/ofis ısıtma ve soğutma senaryosu gerçeklenmiştir. Ev ve ofis ortamlarında ısıtma, soğutma ve güvenlik çözümleri için kablosuz pilli çözümlerin yaygınlaşmaya başladığı düşünülürse, bant verimli ve düşük güç tüketimli çözümlere olan ihtiyaç giderek artmaktadır. Tez çalışmasında önerilen altyapı ile bu ihtiyaca da cevap vermek mümkün olacaktır.

1. INTRODUCTION AND LITERATURE REVIEW

Currently Internet of Things (IoT) concept is still at early stages of development in the industry, it is obvious that it will pervasively come true and billions of embedded microcomputers will become online for the purpose of remote sensing, actuation and sharing information. According to the estimations from academicians and industry leaders, there will be 50 billion connected devices or "things" by the year 2020 [4] [5]. As researchers are developing first to market products, we have chance to identify design parameters, define technical infrastructure and make an effort to meet scalable system requirements. In this manner, required research and development activities must involve several research directions such as massive scaling, robustness, security, openness and human-in-the-loop [6]. Under these circumstances, key research topics for the future will be to realize these future networks which will meet the requirements of the expected growth of IoT [7].

Recently, the explosion in the number of mobile and wireless devices made network society feel the need of different approaches. With the growth of IoT, ensuring the connectivity of devices in local area environments will become more significant and undeniable fact. Especially local area wireless networks will suffer from this large growth when vast amounts of IoT devices become in use in the same vicinity such as houses, offices and production facilities. Furthermore these IoT devices with minimum hardware, software and intelligence will have limited resilience to any imperfections in communication infrastructure. Taking into account all of those needs, we decided to research and study on a different approach, network coding in Heterogeneous Network (HetNet) for the device side communication infrastructure in personal area network. A HetNet is a network in which there are several nodes supporting different communication PHYs simultaneously. The ZigBee-WiFi IP Gateway node developed in this study can be interpreted as a HetNet node.

1.1 Purpose of Thesis

When we think of IoT applications in local area networks, we can't ignore existing and available communication infrastructures and technologies which are generally de facto standards and already deployed. WiFi and ZigBee are two of those communication protocols, so any development on those protocols can be easily adaptable to existing systems.

As the number of wireless sensors and IoT devices increase, unlicensed ISM bands become more and more crowded in radio transmission manner. ISM band systems which uses the same portion of radio spectrum such as WiFi, ZigBee and Z-Wave will encounter more delay, collision and drops due to low SNRs, adjacent and co-channel interferences. As a star topology networking standard, IEEE 802.11a/b/g/n/ac protocols are incapable of connecting more than 15-20 devices per a consumer type combo AP inherently. Moreover, coverage problems due to the nature of star topology and attenuating 2.4 GHz signals will make it impossible to connect distant IoT sensors and devices. For those reasons, mesh topology networking standards such as ZigBee, Z-Wave and IEEE 802.11s come to our minds as a solution to some extent. At the end of the day, all of them will use the same limited frequency spectrum and we need different concepts and new suggestions.

There are ongoing studies at a level of physical layer such as spread spectrum adaptive frequency hopping, antenna diversity and new coding techniques over those. All of them aims to increase finally the quality of service in congested networks. While physical layer improvements take place, upper layer protocols also struggle to guarantee the continuity of data exchange. In this thesis study, we aimed to propose a new method for the reliability, scalability and energy efficiency of LR-WPAN networks. For the ZigBee network to exchange packets with IP networks, WiFi-ZigBee gateway node is planned to be implemented. Network coding will be applied over the existing IEEE 802.15.4 based ZigBee standard for the routing of the traffic between nodes. As a high capacity alternative to conventional routing approaches, network coding techniques provide more robust and efficient communication between nodes by processing the received packets in the intermediate nodes instead of simply storing

and forwarding within the scope of network layer. Thus, network coding provides increased multicast transmit capacity with lower energy consumption. The energy consumption aspect will become significant for battery powered IoT devices in the near future.

1.2 Hypothesis

In 2020s, 50 billion wireless devices will be in the world, indeed most of them will be in our homes, cars, works and body area environment. All of them will use the same limited band, struggling to find uncongested frequency or time slot to transmit information in a selfish way, delayed or lost packets in the chaotic radio spectrum. Even today, we suffer from wireless connectivity problems in our office or home and we often experience low throughput connection performance even there is free up-link and down-link throughput limit. In Figure 1.1 and 1.2 congested 2.4 GHz spectrums of ITU-WCRL on November 21, 2014 and in a hotel in St.Julians, Malta [2] on September 22, 2014 are shown.

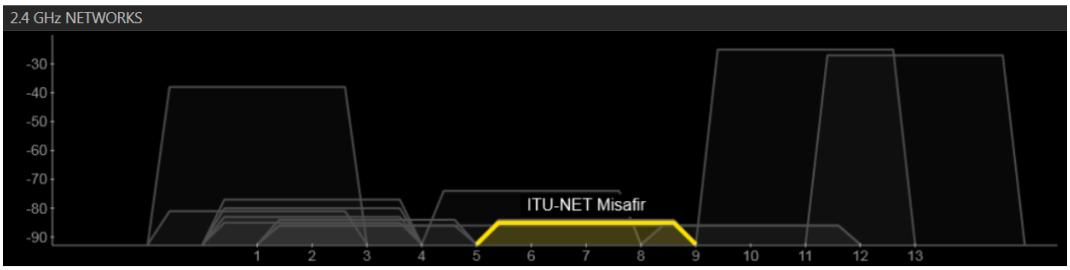


Figure 1.1: Congested 2.4 GHz WiFi spectrum in ITU-WCRL.

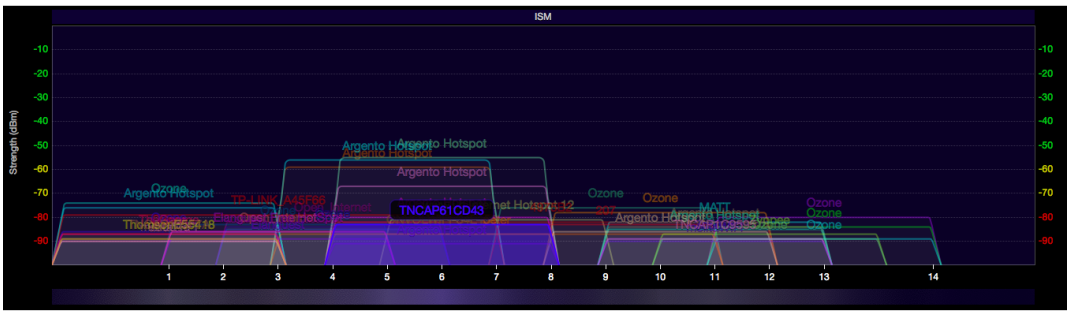


Figure 1.2: In a hotel room in St.Julians, Malta on September 22, 2014 [2].

As the population will become more condensed to constricted high-rise buildings and with the explosion of high bandwidth media applications, electromagnetic emission in unit volume will increase. This situation will act like a natural jamming effect

for each and every wireless device, blocking wireless connectivity, increasing power consumption due to the receiver and transmitter current increase and reducing battery life.

Since ISM band is not unlimited, it is crucial to use it in an efficient way for sustainability of wireless revolution. To be able to achieve this, efficient design attitude should start from the most simplest unit to the most complex one in wireless communication nodes. The efficiency in RF transmission is not only about network interface layer but also all of the protocol layers over PHY. Network coding is a promising technique for using radio spectrum efficiently. If we can converge both network coding techniques and service differentiation for different traffic types using hybrid nodes, we can achieve more efficient wireless communication ecosystem.

1.3 Diversity in Wireless Networks

The basic idea of diversity technique is transmitting the information to the receiver node by using more than one relatively uncorrelated physical or logical routes. This physical or logical routes could provide frequency, time, space, polarization or cooperative diversity which improves the reliability of message transmission by alleviating the effects of interference and fading. Cooperative diversity technique, which enables wireless nodes to contribute cooperation for combating fading and interference, become even more important as the number of wireless nodes increase.

One of the cooperative diversity techniques is network coding in wireless networks. Coded cooperation diversity is widespread in cooperation diversity literature but the studies generally use channel coding technique [8]. The potential of network coding in cooperative diversity came to light in later studies such as [9] and [10].

In [9], researchers implemented linear network coding over both wireless network containing a distributed antenna system as well as user cooperation between users. They analyzed diversity gains and diversity orders of different coding scenarios for both distributed antenna system and cooperation diversity between users. The results they showed that distributed antenna system with network coding gave better diversity performance at with lower hardware cost and higher spectral efficiency. They also showed that network coding yields additional diversity when there are multiple users

in the network at the same instant. They concluded that to achieve further improvement in large networks, distributed channel coding can be used under network coding

In [10], researchers proposed a network coding approach for cooperative diversity featuring the algebraic superposition of channel codes over a finite field. In their study the scenario is a network in which two "partners"—Node A and Node B—cooperate are transmitting information to a single destination. Each node transmits locally generated information and relayed information that originated at the other partner. A key observation in their study was Node B already knows Node A's relayed information since it already generated in Node B and can exploit that knowledge when decoding Node A's local information. This provided an encoding scheme in which each partner transmits the algebraic superposition of its local and relayed information. Message decoding at the destination is then carried out by iterating between the codewords from the two partners. It is shown via simulation that the proposed scheme provides substantial coding gain over other cooperative diversity techniques and proved the improvement in diversity order of the transmitted and relayed traffic with the usage of network coding as a cooperation diversity scheme.

1.4 Network Coding

The idea of network coding was first introduced at the beginning of 2000s, now there are lots of studies and implementations [11] [12] [13]. There is a famous example to summarize the idea behind network coding. In the scenario shown in Figure 1.3, Alice and Bob wants to exchange packets via a relay node in between. First Alice sends her packet to relay node and second relay node forwards it to Bob. Third Bob sends his packet to relay node and fourth relay node forwards it to Alice. This data exchange method requires 4 cycles as you see. To examine the case with network coding approach, consider the following scenario. First Alice sends her packet to relay node and second Bob sends his packet to relay node. In the third step, relay node XORs the two packets and transmits the XORed version. Since Alice and Bob know their own packets, they simply XOR the received packet with their own packets and they obtain each other's packet. The packet exchange ends with 3 cycles and saved cycles can be used for new packet transmissions which increases throughput.

Figure 1.3: Conventional (a) vs. network coding (b) approach. 3 transmissions instead of 4 saves 1 transmission cycle for any other data exchange.

Network coding is a promising method for future wireless networks and it has lots of advantages. Some of the most important IoT related advantages of network coding are low complexity for near-optimal routing [14] [15] [16] [13], throughput enhancement [11], robustness to link failures and packet losses [14] [17]. In literature, there are several researches and implementations with the use of network coding in wireless networks. To look at those practical studies and investigate the relations with our study, consider the following reviews.

In [1], researchers proposed a new packet forwarding architecture which they called COPE. The aim was to improve the throughput of the wireless network. They implemented a coding scheme between IP and MAC layers of internet protocol suite. The main idea behind COPE is, it identifies coding opportunities and forwards multiple packets in a single transmission. COPE uses three main techniques: Opportunistic listening, opportunistic coding and learning neighbor state. Opportunistic listening means, each node snoops the wireless medium and stores all the packets around it for a limited time period. Also each node broadcasts reception reports to tell its neighbors which packets they have in their packet pool. Another term opportunistic coding means, finding the best coding option to maximize total throughput. Figure 1.4 summarizes opportunistic coding. Node B has 4 packets in its output buffer, packet nexthops are listed in (b). Each neighbor of Node B has stored some packets as shown in (a). Node B has to choose best coding option which maximizes throughput, this time it is P1+P3+P4 [1].

Lastly, they defined a prevention mechanism, learning neighbor state, for severe congestion situations. When there is severe congestion on the network, nodes may not receive reception reports and so they can't get the information of neighbor packets. This will cause wrong coding decisions and extreme decline in throughput. To be able

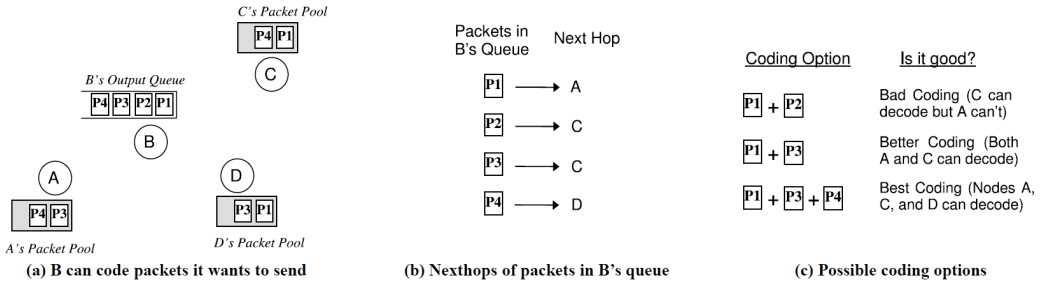


Figure 1.4: Opportunistic coding example [1].

to prevent this, they leveraged ETX metric [18] which the link-state routing protocols utilize to guess delivery probabilities of each link intelligently.

In this study, the researchers mainly focused on throughput so they calculated the theoretical coding gains of COPE for several network topologies. They basically used the ratio, number of transmissions without coding over number of transmissions with coding. Table 1.1 shows coding gains of COPE for several network topologies.

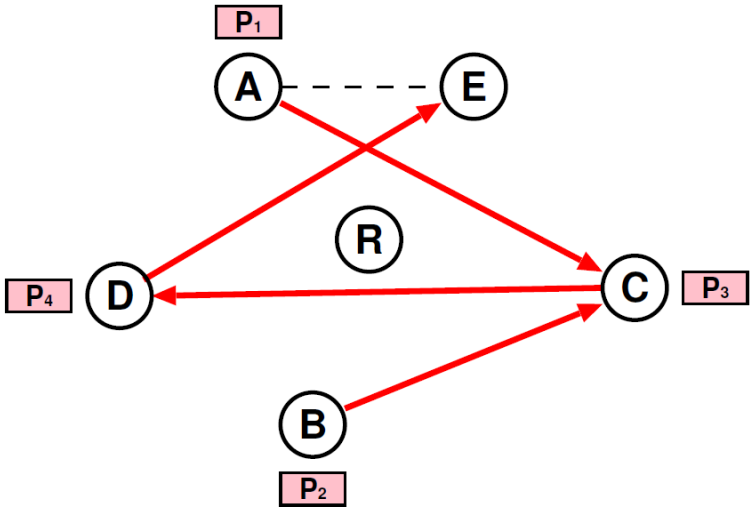
Table 1.1: Theoretical coding gains of COPE for several basic topologies [1].

Topology	Coding Gain	Coding + MAC Gain
Alice-and-Bob	1.33	2
"X"	1.33	2
Cross	1.6	4
Infinite Chain	2	2
Infinite Wheel	2	∞

For the experimental studies, they used 20-node wireless testbed that spans two floors in their department building. Nodes of their testbed ran Linux and they used 802.11a with bit-rate of 6 Mbps. COPE was implemented on the nodes using the Click toolkit [19]. They validated their theoretical calculations with practical results for Alice-and-Bob, X and cross topologies.

In another research for wireless network coding [3], researchers established their studies on [11] and they made comparison with COPE [1] which we discussed earlier. They showed some situations that COPE is not applicable but still useful network coding is actually possible. One of the example scenarios is in Figure 1.5.

When we look at the table in Figure 1.5, first column shows the packets to transmit, second and third columns show the sender and the receiver nodes of the corresponding packets, respectively. The question is, if there is a beneficial coding scheme for this packet transmission knowing that each node has a key set written in the



	sender	receiver	key set
P_1	A	C	$\{P_3\}$
P_2	B	C	$\{P_3\}$
P_3	C	D	$\{P_4\}$
P_4	D	E	$\{P_1\}$

Figure 1.5: The central relay node can communicate with each node. Surrounding nodes can hear each other if there is a dashed line between them [3].

table. According to COPE algorithm, packets P_1, P_3 and P_4 is forwarded by R but broadcasting P_1 P_3 and P_3 P_4 is a useful coding scenario and saves 1 transmission cycle for relay node R. So researchers of [3] stated that COPE is just the subset of network coding.

Not only they showed an example that COPE is insufficient but also they defined the applicability condition for existence of a useful network coding scheme. For this purpose, they first proved the sufficiency of applicability condition and then if the applicability condition is satisfied, they showed that there exists a useful coding scheme, regardless of the network topology. The last but not the least, they defined a class of robust coding schemes, called loop coding which reduces post-coding loss rate.

One of the studies on physical-layer network coding is [20]. In this study, collision coding scheme is used in physical layer for resolving collided ACKs. The study is based on the seminal work on physical-layer network coding (PNC) [21] which

suggests to decode two simultaneous wireless transmissions using XOR principle at the electromagnetic wave level. It has been shown that, for a multicast network with N receivers and a batch size of M packets, it is a NP-complete problem to minimize the number of transmit cycles for linear coding the packets over $GF(2)$ [22] [23]. Then researches stated that there exists lots of studies [24] [1] [25] on heuristic algorithms to minimize number of transmissions using $GF(2)$ linear coding, however the throughput performance significantly reduces under the optimal performance. Although it is possible to optimize throughput by coding over a larger finite field size of $GF(2^q)$ where $q \geq \lceil \log_2 N \rceil$ [23] [26], it has higher delay constraint due to encoding/decoding computational complexities [27], hence encoding/decoding throughput [28].

1.5 Random Linear Network Coding

The core idea of random linear network coding is to allow scrambling data at intermediate nodes [29] by generating a linear combination of the incoming packets using random encoding coefficients. For dynamic networks in which the node connections are varying over time, random network coding can achieve coding and encoding packets with a high probability of recovery. Due to this characteristic feature, random linear network coding is suitable for wireless mesh networks where the connections between nodes are varying over time.

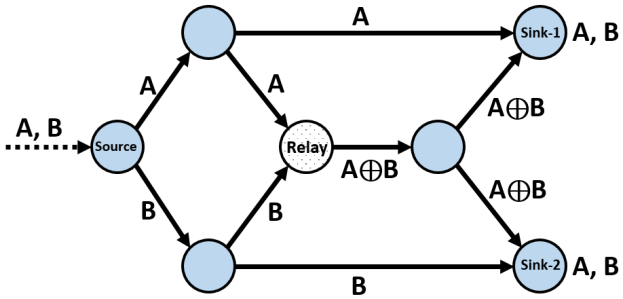


Figure 1.6: The butterfly network. Each directed link in this network is capable of transferring a single packet in one cycle. There are 2 data packets, a and b, which must be transferred to the Sink-1 and Sink-2 nodes.

Consider the butterfly network shown in Figure 1.6. For the relay node in between, simply addition of the bits of incoming packets in $GF(2)$ for coding purposes means bitwise XOR. For practical consideration, the Relay node in the middle uses XOR operation for encoding the packets while Sink-1 and Sink-2 nodes use XOR operation for decoding the packets. This is a good example for the operation of linear network

coding. When we talk about random linear network coding, the network in Figure 1.7 makes more sense, since the number of relay nodes are more than the number of source nodes. In this network, random coding vector coefficients are used to encode the packets in relay nodes and recover the encoded symbols in the destination node.

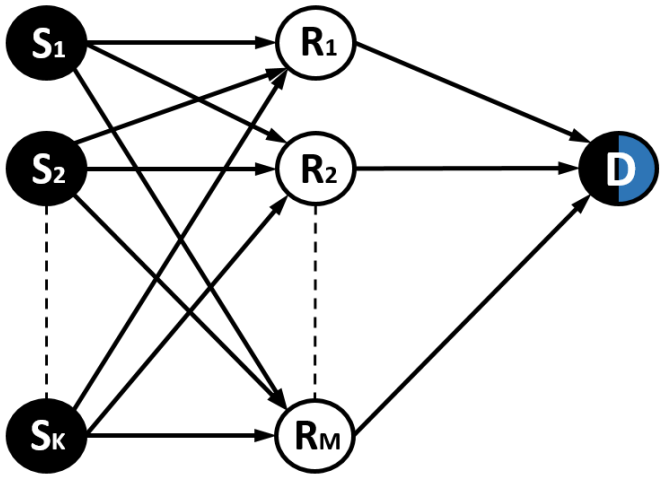


Figure 1.7: Wireless mesh network with source nodes S_i, relay nodes R_j and a destination node D. Relay nodes in between source nodes and destination node have capability to encode the packets by using random vectors.

1.6 Contributions

Different from conventional routing approaches, network coding is a new and promising technique in wireless communication applications including new concepts such as IoT and wireless sensor and actuator networks (WSAN). In this study, we aimed to converge network coding applications with existing and available technologies in the industry. By doing this, we proposed theoretical construction of the network coding techniques over IoT and WSAN applications. We also realized those concepts into a real and working testbed environment which we implemented during the study.

Our first and most important practical contribution is implementing end-to-end testbed including service oriented [30] [31] cloud server application on a portable virtual server OS and its client application on a limited-featured ARM Cortex-M3 embedded node. Thus a simple but all-purpose platform for WSAN is provided for wide range of applications in IoT concept. This platform will pave the way for different studies and applications within the scope of ITU-WCRL research topics.

Our second contribution is implementing a WiFi-ZigBee gateway that enables ZigBee LR-WPAN sensors and actuators to connect through or to be reached from IP networks. For this development, both hardware and software implementations are completed on Broadcom WICED WiFi and TI CC2530 ZigBee platforms. Since those platforms are not only available in the market but also state-of-the-art technologies, the project deliverables and useful concepts can be easily adapted to the industrial applications.

Our third contribution is implementing network coding (XOR coding) software application for both gateway node and LR-WPAN sensor and actuator nodes. Although XOR coding is a very simple network coding application, this development forms the basis for more advanced network coding studies such as random network coding on our testbed environment.

2. NETWORK CODING: THEORY AND SIMULATIONS

This chapter contains theoretical studies on network coding in low-rate wireless personal area networks (LR-WPAN). The network coding operation for IP gateway node and packet decoding and recovery problem for end-devices are investigated in detail. Furthermore, random linear network coding simulations are developed in Matlab environment for testing the performance of network coding techniques in LR-WPAN.

2.1 Random Linear Network Coding Theory and Practice

To visualize random network coding in a mesh network, we have to define a packet transmission scenario from source node to destination node. Consider the network in Figure 1.7 which is introduced in Chapter 1. On the left hand side there are source nodes, in the middle there are relay nodes and on the right hand side there is a destination node. Note that the number of relay nodes M must be greater or equal to the number of source nodes K ($M \geq K$), to ensure destination node to collect sufficient number of coded packets for recovery.

Consider original packets $\mathbf{a_i}$ from source nodes $\mathbf{S_i}$ where $i = [1,k]$ in Figure 1.7. These source nodes are transmitting packets to the destination node $\mathbf{D}$ via relay nodes $\mathbf{R_j}$ where $j = [1,m]$

$$\overrightarrow{\mathbf{a}} = \{a_1, a_2, ..., a_k\} \tag{2.1}$$

During the packet forwarding operation at the relay nodes $\mathbf{R_j}$, each packet from source nodes $\mathbf{S_i}$ are encoded using the random coefficients $\mathbf{G_{ij}}$ where $i = [1,k]$ and $j = [1,m]$.

$$\overrightarrow{\mathbf{G}} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & . & . & \alpha_{1k} \\ \alpha_{21} & \alpha_{22} & . & . & \alpha_{2k} \\ . & . & . & . & . \\ . & . & . & . & . \\ \alpha_{m1} & \alpha_{m2} & . & . & \alpha_{mk} \end{bmatrix} \tag{2.2}$$

The destination node **D** receives the additive random coded message vectors $\mathbf{b_j}$ where $b_j = \sum_{l=1}^{K} \alpha_{jl}.a_l$ and $j = [1,m]$. The overall operation can be summarized using the following equation:

$$\begin{bmatrix} b_1 \\ b_2 \\ . \\ . \\ b_m \end{bmatrix} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & . & . & \alpha_{1k} \\ \alpha_{21} & \alpha_{22} & . & . & \alpha_{2k} \\ . & . & . & . & . \\ . & . & . & . & . \\ \alpha_{m1} & \alpha_{m2} & . & . & \alpha_{mk} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ . \\ . \\ a_k \end{bmatrix} \tag{2.3}$$

Consider the packet recovery problem from receiver side, denoting the received packet matrix $\overrightarrow{\mathbf{R}}$ as:

$$\overrightarrow{\mathbf{R}} = \begin{bmatrix} \alpha_{11}a_1 & \alpha_{12}a_2 & . & . & \alpha_{1k}a_k \\ \alpha_{21}a_1 & \alpha_{22}a_2 & . & . & \alpha_{2k}a_k \\ . & . & . & . & . \\ . & . & . & . & . \\ \alpha_{m1}a_1 & \alpha_{m2}a_2 & . & . & \alpha_{mk}a_k \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ . \\ . \\ b_m \end{bmatrix} \tag{2.4}$$

Applying Gauss - Jordan elimination method to the received packet matrix, coding matrix $\overrightarrow{\mathbf{G}}$ is obtained and original packets $\mathbf{a_i}$ are recovered in the destination node using the following equation:

$$\overrightarrow{\mathbf{a}} = (\overrightarrow{\mathbf{G}})^{-1}\overrightarrow{\mathbf{b}} \tag{2.5}$$

2.2 LR-WPAN Random Linear Network Coding Simulations

The benefits of random linear network coding in low-rate wireless personal area networks are examined by the developed simulations in Matlab environment as part of this thesis study.

Basically, the network in Figure 1.7 is used to simulate RLNC activity between nodes. There are three basic types of nodes in the network: Source nodes, relay nodes and a destination node. Source nodes are the data generators of the network. Each source node creates a payload data and transmits it to the near relay nodes. Since the network is a mesh type network, each relay node close to the source nodes receives the packets. When there are more than one packet from different source nodes, it becomes possible to code the packets and send coded packets to the destination node.

The simulations are based on the symbol erasure fact of wireless networks. Due to the interference from radio transmitters, ISM band technologies frequently encounter symbol erasures and it is very important to be able to recover actual payload in case of a symbol erasure during transmission. Retransmissions caused by symbol erasure result in not only throughput decreases but also excessive energy consumption which is critical for battery powered sensor devices.

The development of the simulations are based on Galois Field arithmetics and random number generation functions of the Matlab environment. The simulation applications are developed dynamically to be able to simulate different types and sizes of networks. Random linear network coding symbol decoding and recovery property is examined for a range of channel erasure probabilities and the results are smoothed using Monte Carlo method by increasing the length of the transferred data. At the end of the simulation, the results of different Galois Field sizes are compared and validated by using the results from uncoded bare message transfer scenario.

The following part analyzes random linear network coding simulation results of the networks with different number of nodes. In Figures 2.1 and 1.7, each source node consecutively send its payload to the relay nodes and relay nodes transfers data after coding it with a random generator vector. The destination node then receives the coded messages and checks if it is possible to decode by Gauss - Jordan elimination. Considering Gauss - Jordan elimination and packet recovery problem mentioned in Section 2.1, the number of relay nodes are always greater or equal to the number of source nodes.

As depicted in Figure 2.1, one of the simulated network scenarios have 2 source nodes, 3 relay nodes and a destination node in a mesh connectivity. Relay nodes in between uses random vectors to code the packets while forwarding the them to the destination node.

In Figure 2.2, channel erasure probability vs. symbol error probability graph of the network coding scenario in Figure 2.1 is shown. Each colored line represents different results of symbol error probability for different Galois Field power $m = [1,6]$ in the channel erasure probability range $p = (0,1)$.

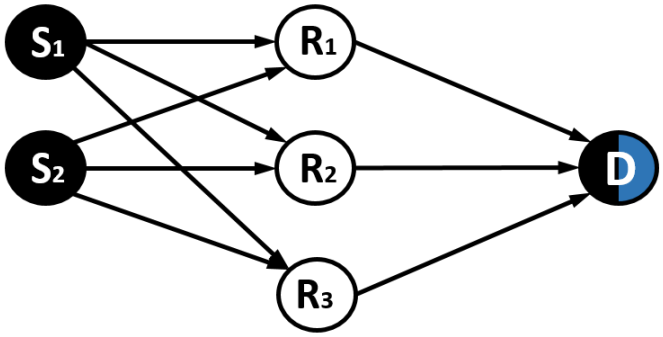


Figure 2.1: Wireless mesh network with 2 source nodes, 3 relay nodes and a destination node. Relay nodes in between source nodes and destination node have capability to encode the packets by using random vectors.

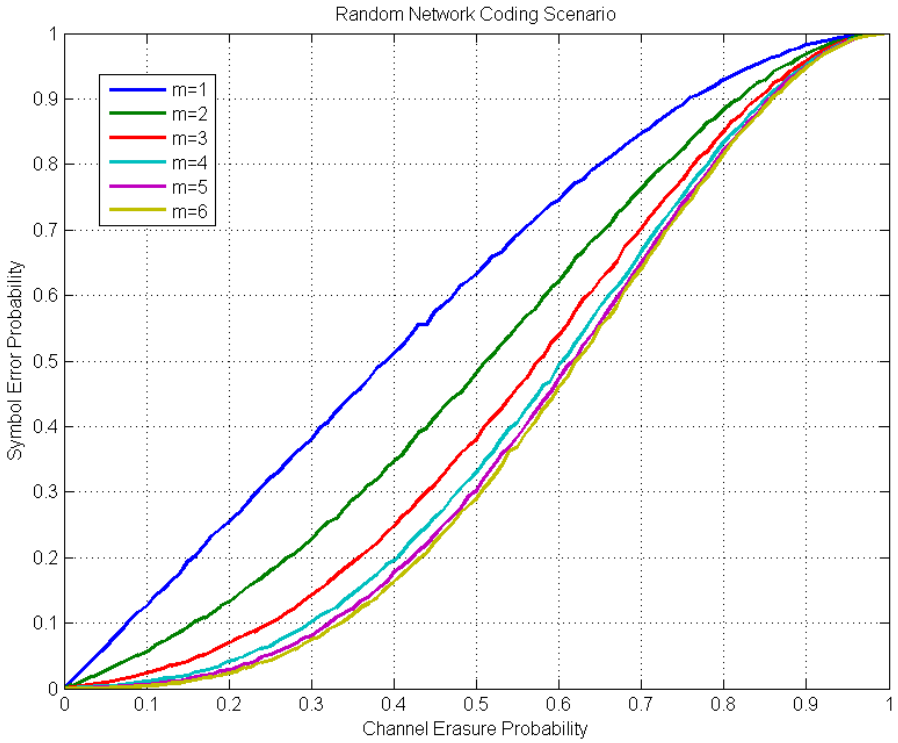


Figure 2.2: Channel erasure probability vs. symbol error probability graph of the network in Figure 2.1. Different colored graphs represent different Galois Field power 2^m of m from 1 to 6.

To be able to verify the performance of network coding, channel erasure probability vs. symbol error probability graph of uncoded message transfer scenario is depicted in Figure 2.3. It is easy to notice that, starting from $GF(2^2)$ case in Figure 2.2, symbol error probabilities of higher Galois Field powers are better than the uncoded message transfer scenario.

To increase coding rate, also the network in Figure 2.4 with 5 source nodes, 6 relay nodes and a destination node is simulated. The results verified that the increase in

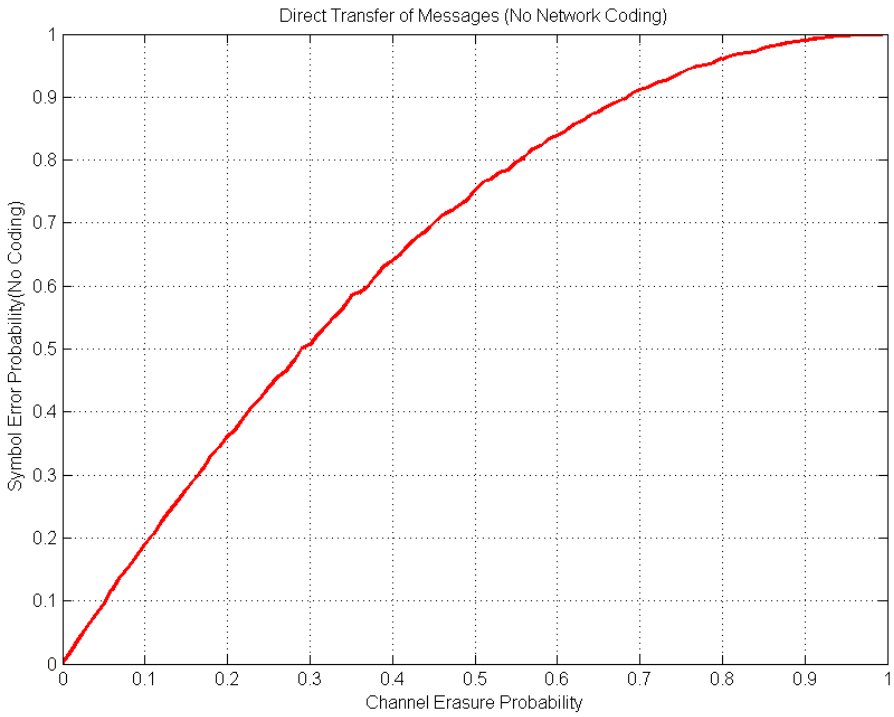


Figure 2.3: Channel erasure probability vs. symbol error probability graph of the network in Figure 2.1 for uncoded bare message transfer scenario.

coding rate provides lower symbol error probabilities for the same level of channel erasure probabilities.

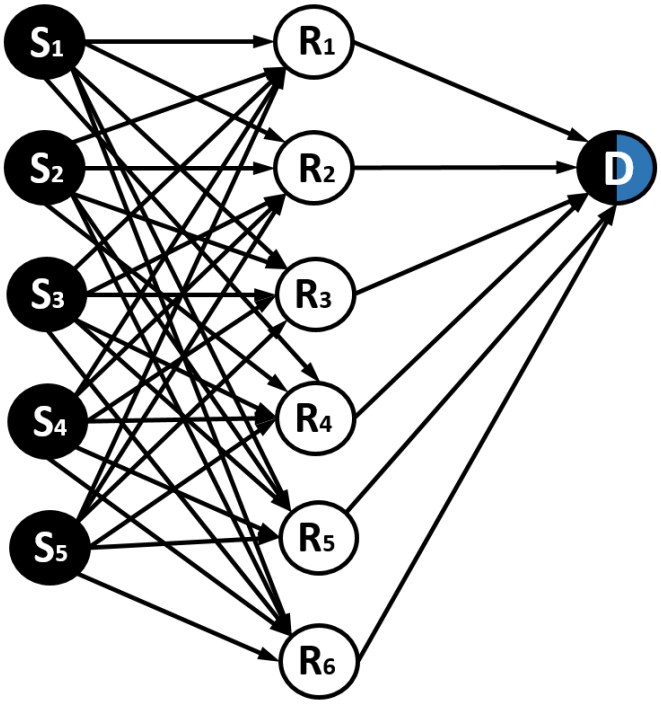


Figure 2.4: Wireless mesh network with 5 source nodes, 6 relay nodes and a destination node. Relay nodes have capability to encode the packets by using random vectors.

Similarly in Figure 2.5, channel erasure probability vs. symbol error probability graph is shown. Each colored line represents different results of symbol error probability for different Galois Field power $m = [1,6]$ in the channel erasure probability $p = (0,1)$.

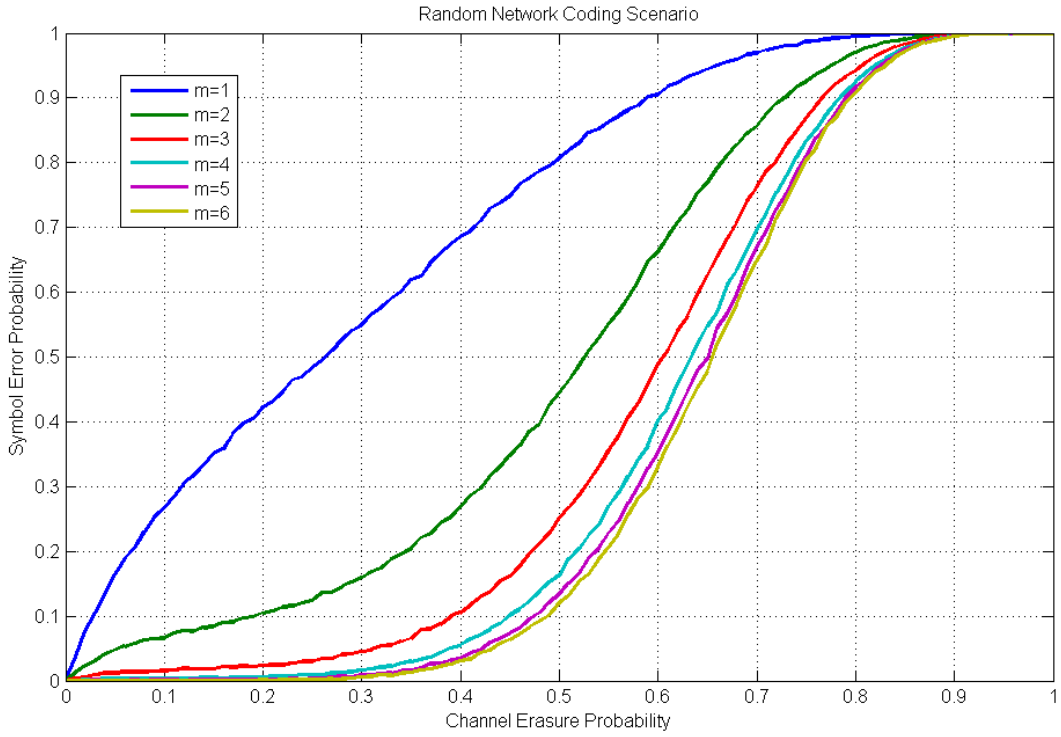


Figure 2.5: Channel erasure probability vs. symbol error probability graph of the network in Figure 2.4. $m = 1$ to 6 represent different Galois Field powers.

Also in Figure 2.6, uncoded message transfer scenario result is shown for a channel with channel erasure probability $p = (0,1)$.

As we extend the coding simulations by limiting the coding vectors to nonzero random coefficients, we obtain better performance results. In Figure 2.7, dark blue and red lines are the results of $GF(2^2)$ and $GF(2^3)$ random coding coefficients respectively without nonzero limitation. Green and turquoise lines are the results of $GF(2^2)$ and $GF(2^3)$ random coding coefficients respectively with nonzero limitation. Since decoding performance increases by choosing better random coding coefficients, we observed this improvement from symbol error probability graph implicitly.

Additionally in Figure 2.8, random coding coefficient results of $GF(2^4)$, $GF(2^5)$ and $GF(2^6)$ are shown with and without nonzero limitation. It is obviously seen that nonzero coding coefficients performed even better than higher powers of GF and in other words larger coding coefficient sets.

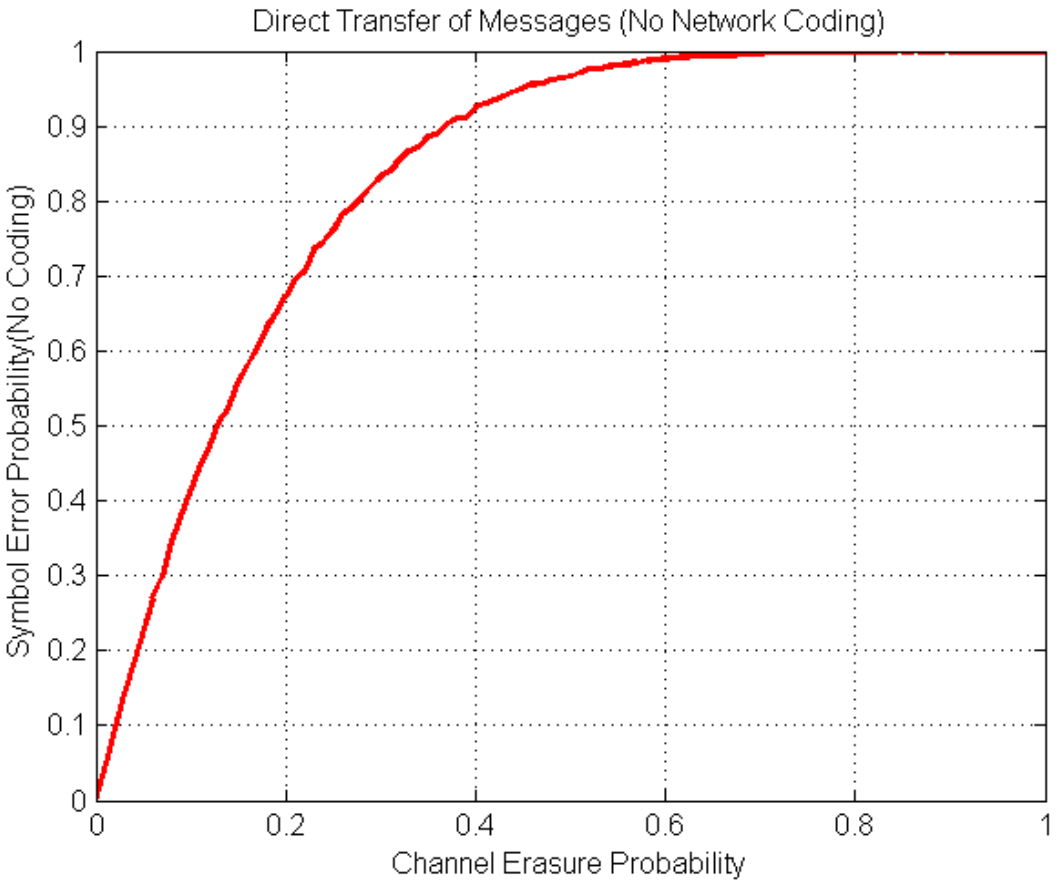


Figure 2.6: Channel erasure probability vs. symbol error probability graph of network in Figure 2.4 for uncoded bare message transfer scenario.

During the development and simulation phases; memory preallocation techniques and parallel programming features of Matlab environment are used to be able to decrease the duration of simulation runs. Simultaneously running Matlab sessions are used for parallel programming features and parallel computations during the simulations. 3 to 5 times reduction is achieved in simulation durations. The algorithm and Matlab code of random linear network coding simulations can be found in Appendix 1.1.

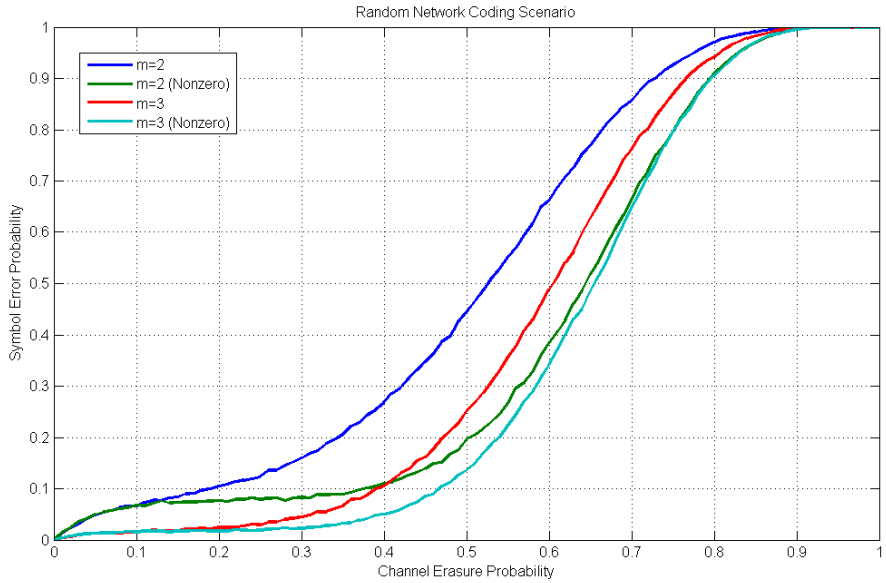


Figure 2.7: Nonzero coding coefficient performance improvement for 5 source, 6 relay network with random network coding. Dark blue and red lines are the results of random coding coefficients from $GF(2^2)$ and $GF(2^3)$ without nonzero limitation while green and turquoise lines are the results of random coding coefficients from $GF(2^2)$ and $GF(2^3)$ with nonzero limitation.

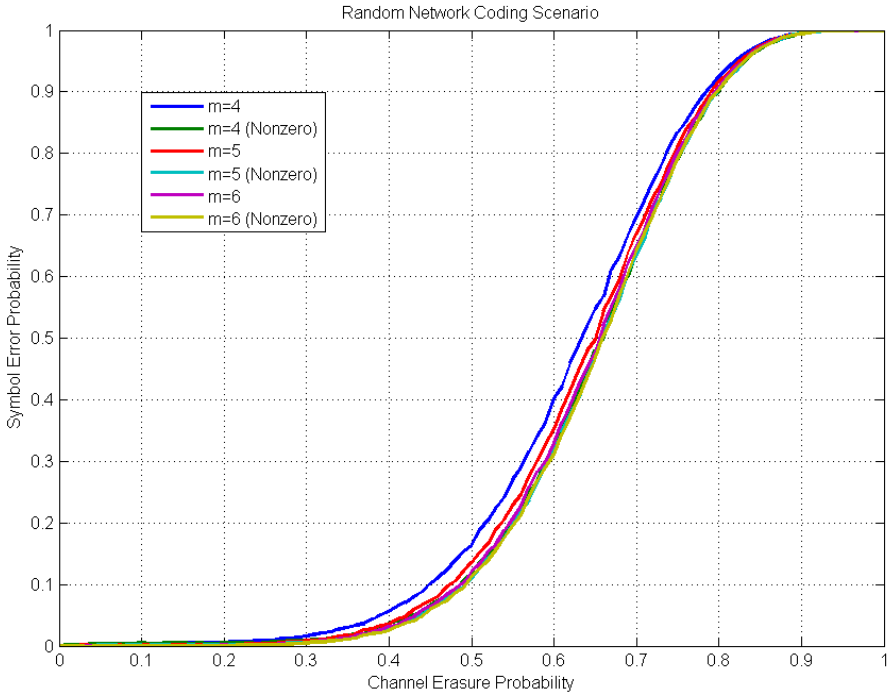


Figure 2.8: Nonzero coding coefficient performance improvement for 5 source, 6 relay network with random network coding. Random coding coefficient results are shown with and without nonzero limitation from $GF(2^4)$, $GF(2^5)$ and $GF(2^6)$.

2.3 Conclusion of Simulations

In conclusion, the performance of random network coding for different exponential powers m of $GF(2^m)$ is investigated for different channel erasure probabilities. Simulation results are combined in the graphs, to be able to compare and cross check different values of channel erasure probabilities and GF sizes. Results are smoothed using Monte Carlo technique by using increased data length. At the end of the simulations, the performance of random network coding in LR-WPAN with mesh capability is validated.

3. BACKGROUND KNOWLEDGE ON APPLIED CONCEPTS

This chapter contains background knowledge about basic concepts that are utilized during the study. Some of the network protocols, standards and techniques are in the scope of this chapter. All of the topics are summarized for giving only sufficient information, not to overload the reader.

3.1 ZigBee Protocol and Network Architecture

ZigBee is a specification build upon IEEE 802.15.4 LR-WPAN standard [32]. IEEE 802.15.4 standard defines the PHY and MAC layer for a low data rate, low complexity devices with long battery life. ZigBee defines network specifications and framework for application layer over IEEE 802.15.4 MAC. Figure 3.1 shows the layered communication architecture of ZigBee.

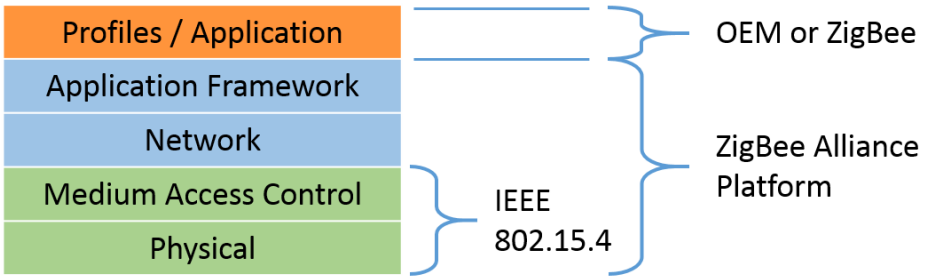


Figure 3.1: ZigBee layered architecture.

Some of the features provided by IEEE 802.15.4 standard are as follows:

- Star network or peer-to-peer operation
- Unique 64-bit or allocated 16-bit addressing scheme
- CSMA-CA channel access
- Full acknowledgement for transfer reliability
- Low power consumption

- Energy detection and link quality indication

This standard defines 2 types of devices called full-function device (FFD) and reduced-function device (RFD). An FFD serves as the PAN coordinator and an RFD functions as the simple sensor device. ZigBee coordinator and router are FFDs while ZigBee end-device is an RFD in IEEE 802.15.4 standard.

IEEE 802.15.4 LR-WPAN, and so ZigBee, can operate in either star or peer-to-peer topology. According to the requirements, one can define the topology by using FFDs and RFDs in the application environment. Independent of the topology, each ZigBee PAN must have its own PAN coordinator device which forms the network with a PAN identifier that is not currently used by another PAN within the radio range. Once the coordinator forms the network, it allows other FFDs and RFDs, to join the network. The basic structure of ZigBee star and peer-to-peer topologies are illustrated in Figure 3.2.

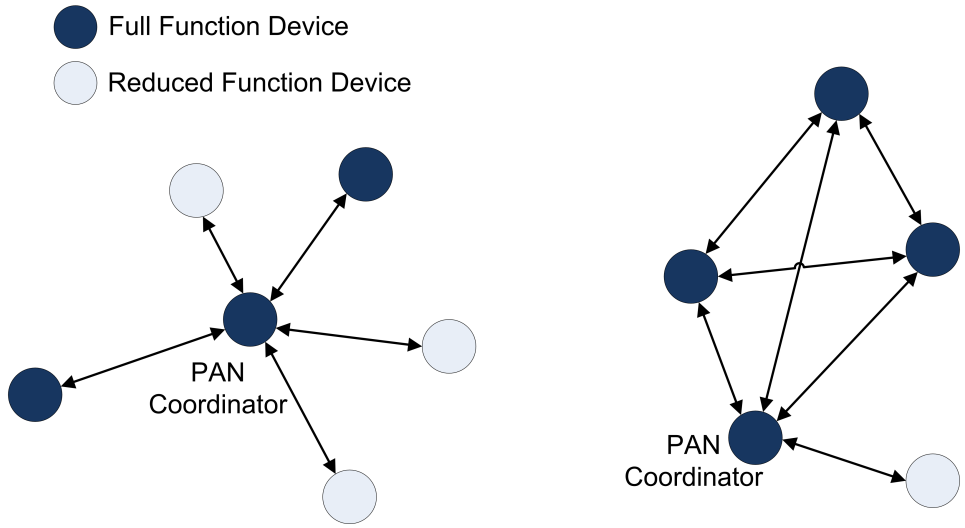


Figure 3.2: ZigBee star and mesh network topology examples.

IEEE 802.15.4 PHY: There are two ZigBee PHYs based on IEEE 802.15.4 standard, 868/915 MHz and 2450 MHz. Table 3.1 summarizes the characteristics of ZigBee PHY layers.

There are 27 frequency channels in which ZigBee devices operate. 11 of these channels are in 868/915 MHz PHY and 16 of these are in 2.4 GHz PHY. Adjacent channel separation is 2 MHz for 915 MHz band and 5 MHz for 2.4 GHz. Figure 3.3 shows 868/915 MHz and 2.4 GHz band ZigBee channels.

Table 3.1: IEEE 802.15.4 based ZigBee PHY layer

PHY (MHz)	Frequency band (MHz)	Spreading parameters		Data parameters		
		Chip rate (kchip/s)	Modulation	Bit rate (kb/s)	Symbol rate (ksymbol/s)	Symbols
868/915	868–868.6	300	BPSK	20	20	Binary
	902–928	600	BPSK	40	40	Binary
2450 DSSS	2400–2483.5	2000	O-QPSK	250	62.5	16-ary orthogonal

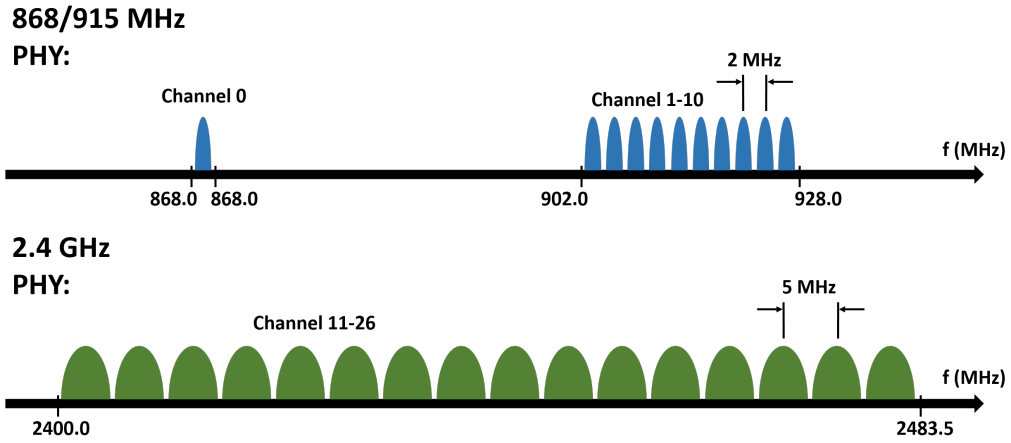


Figure 3.3: ZigBee PHY layer frequency bands.

ZigBee PHY frames are created between 6 to 133 octets. First 5 octets are for synchronization purposes. Respectively, 4 octets preamble and 1 octet start of packet delimiter (11100101). After synchronization header, 1 octet PHY header and 0-127 octets PHY Service Data Unit comes. PSDU part is payload from 802.15.4 MAC layer. Figure 3.4 illustrates IEEE 802.15.4 PHY frame.

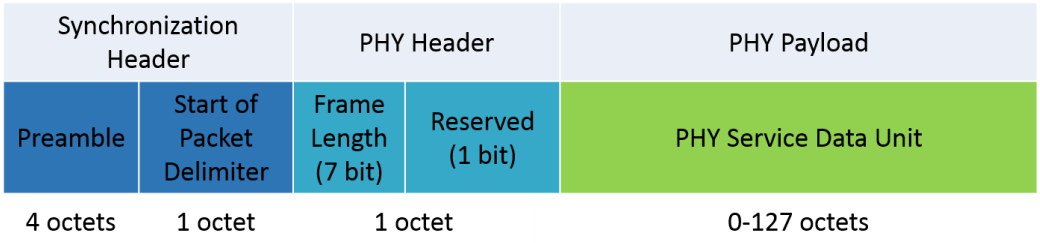


Figure 3.4: ZigBee PHY layer frame format.

IEEE 802.15.4 MAC: ZigBee uses MAC sublayer of IEEE-802.15.4 LR-WPAN standard. The MAC sublayer is responsible for the following tasks in a ZigBee node [32] [33]:

- Generating network beacons if the device is a coordinator

- Synchronizing and locking on network beacons
- PAN association and disassociation
- Realizing CSMA-CA mechanism
- Providing reliable link between peers
- Securing communication between devices
- Handling guaranteed time slot mechanism for channel access

ZigBee MAC layer frames consist of MAC header, MAC Service Data Unit and MAC footer. This is the general frame format that ZigBee MAC layer uses. Figure 3.4 illustrates IEEE 802.15.4 MAC frame. This frame is a payload for PHY layer.

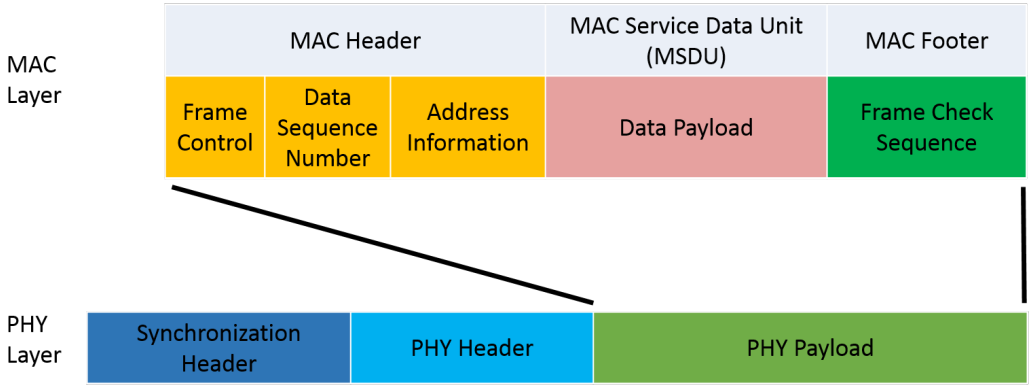


Figure 3.5: ZigBee MAC layer frame format.

IEEE 802.15.4 MAC layer has different MAC frames for different purposes. For example in star network topology, Beacon MAC frame is used for node synchronization and PAN broadcast. For upper layer payload, data frame type is used etc. There are 4 types of MAC frames:

1. Data Frame
2. Beacon Frame
3. Acknowledgment Frame
4. MAC Command Frame

3.2 TCP/IP

The Internet protocol suite, known as TCP/IP, is the set of communication protocols which defines the computer networking architecture. TCP/IP defines end-to-end connectivity and transmission of data over networks. There are 4 abstraction layers in TCP/IP model: Application, Transport, Internet and Network Interface. Figure 3.6 shows TCP/IP layered architecture and OSI reference model equivalent of each layer.

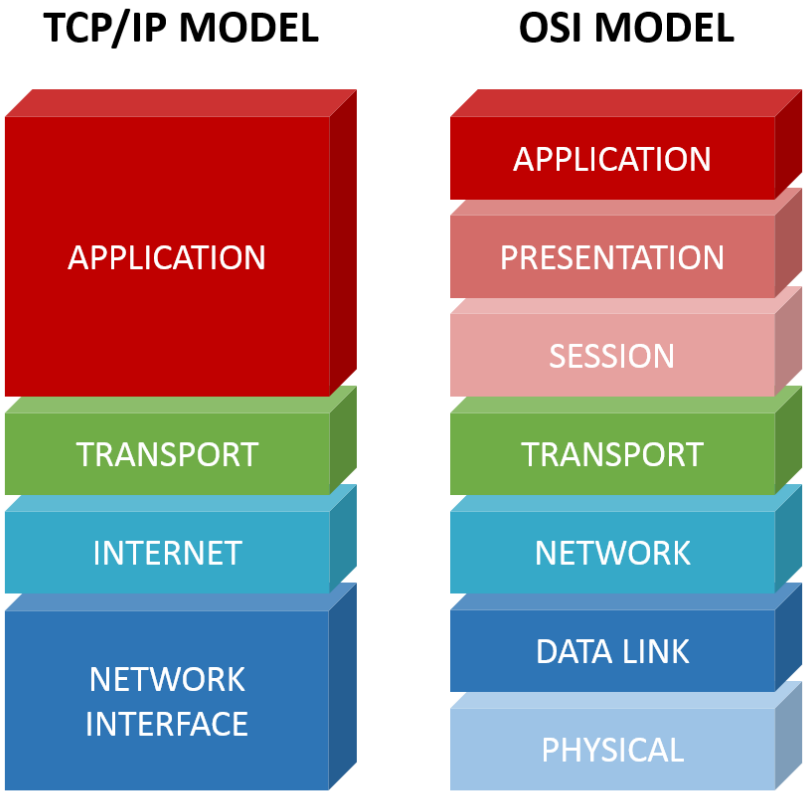


Figure 3.6: TCP/IP and OSI reference models.

In OSI reference model, the architecture was defined in detail where each layer had its own protocol. For example, application were isolated from presentation and session layers. Likewise, data link layer was separate from physical layer. Today, it has become hard to split layers from each other using new protocols. For example, the application layer protocols used in computer communications today combines application, presentation and session layers in OSI model. TCP/IP model is now sufficient to demonstrate each step for a transmission of data between two peers. In Figure 3.7, Peer A sends data to Peer B. The encapsulation and decapsulation

procedures in layered model of TCP/IP are demonstrated in detail. First Peer A's application layer protocol creates a payload and transfers it to the transport layer protocol. Transport protocol adds transport header to the data and forms a segment. It transfers this segment to the internet layer protocol. Internet layer protocol adds internet header and forms packet, then transfers it to the network interface layer. Last, network interface protocol adds header and trailer to this packet and creates frame. This process from application layer to network interface layer is called encapsulation. Now data is ready to be sent over a transmission medium. Frame is transferred from Peer-A to Peer-B through the transmission medium. After Peer-B receives the frame using its network interface layer, it applies exactly the inverted procedure and decapsulates the frame from its network interface layer to its application layer. Now, application layer protocol of Peer-B receives the data and processes it.

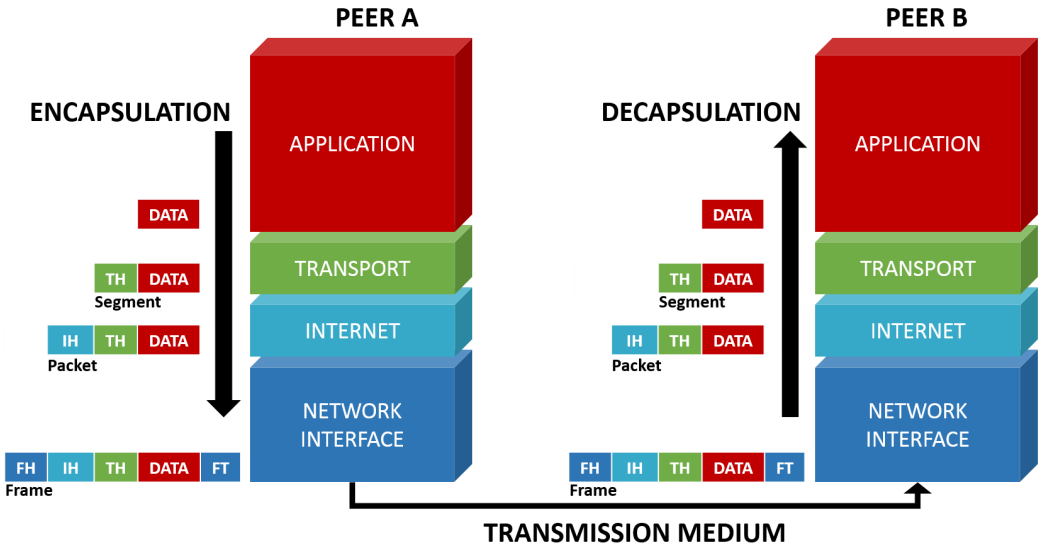


Figure 3.7: TCP/IP encapsulation and decapsulation procedures.

3.3 Real Time Operating Systems

One of the major elements of an embedded applications is its software architecture. Embedded systems often comprise a real time operating system. An RTOS must be consistent in terms of the amount of time it takes to respond to processing demands and to complete the related task. In short, an embedded system must behave deterministically in meeting deadlines set by specific applications and external events. [34]

3.3.1 Scheduling Basics

In simple embedded system applications, interrupt service routines and main operation loops are used for balancing processing capacity between real time events and data processing purposes. However, even the chosen system processing capacity is utilized about % 50 level, handling relatively long events prevent processing of other events at regular time intervals. Table 3.2 summarizes the situation by giving an example processing time calculation of a simple embedded system without an RTOS.

Table 3.2: Sample timing specifications for a simple embedded system.

Tasks	Time per single run	Maximum frequency	Process time for the task in 1s
ISR-1	1 ms	10/s	10 ms
ISR-2	1 ms	4/s	4 ms
ISR-3	1 ms	1/s	1 ms
Event-1 data process	10 ms	10/s	100 ms
Event-2 data process	20 ms	2/s	40 ms
Event-3 data process	300 ms	1/s	300 ms

As it can be seen from Table 3.2, total process time required for all tasks in 1s of the CPU is 455 ms. Although the processor is capable of handling at least twice as much the required load, it won't be possible to process event-1 and event-2 data for required time intervals because of event-3. ISRs remain to be called since all the processing capacity is dedicated with predefined priority when there is an interrupt. The solution of this problem is to interrupt event-3 also for processing of event-1 and event-2 data. This simple idea forms the basis of all RTOS and multi-thread processing software architectures. In Figure 3.8, processor operation using basic RTOS schedular with 100 ms timer interrupt is shown.

There are several scheduling strategies for processor time distribution over concurrent tasks on a processor:

- First In First Out
- Shortest Remaining Time
- Fixed Priority Scheduling
- Round-Robin Scheduling

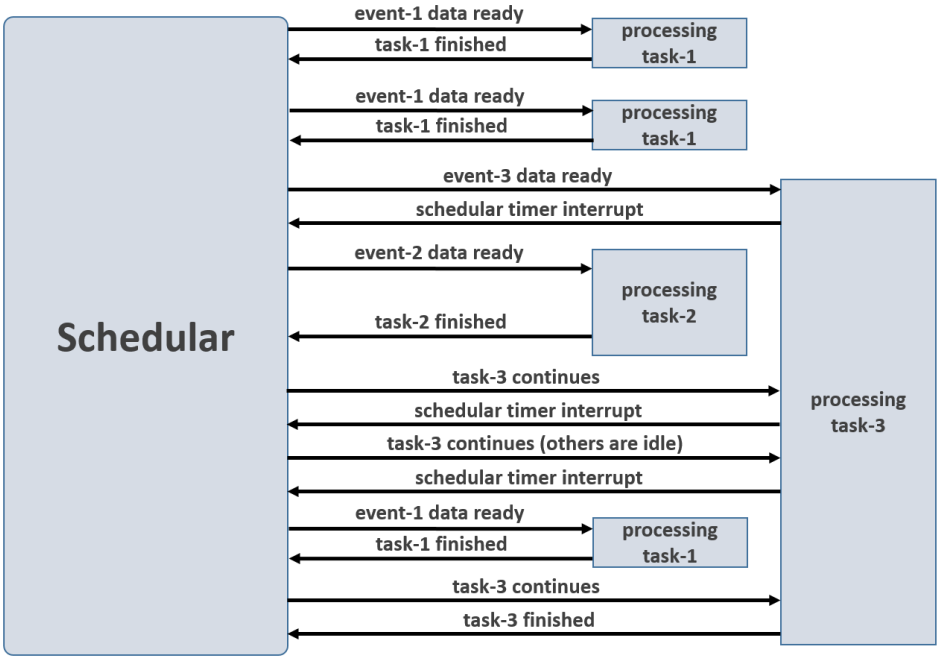


Figure 3.8: Sequence of events occurred in an embedded system with timing specifications defined in Table 3.2.

- Multilevel Queues

3.3.2 Threads

For single core CPU architectures, concurrent execution of several processes is realized using scheduling algorithms. This CPU switching between different threads is time division multiplexing technique. Threads in operating systems are the parts of the programs which are concurrently executed by the CPU. In RTOS software architecture, every process has a thread which executes the event related with that process. Either using the same threads for different events or using different threads for different events is possible for distributing processing power and planning system resources.

3.4 Conclusion of Background Knowledge

In this chapter, general information about applied concepts are provided to the reader of the thesis. The aim is to speak the same language with the reader and prepare the reader to comprehend the basic structure of the study. Further and fundamental information can be found from reference publications and books in the light of given information in this chapter.

4. HARDWARE DEVELOPMENT

Wireless networks and devices are the main focus field of this study. Since ZigBee and Wi-Fi protocols are chosen as the heterogeneous network components in the proposed system, several radio equipments and modules are used to implement device side network infrastructure. Device side network nodes contain both ZigBee and WiFi transceivers separately, making it possible to communicate on ZigBee and WiFi networks at the same time. Although it is possible to use only one RF front-end for both 2.4 GHz ZigBee and WiFi RF communication, these kinds of chips are not widespread in the market and it is better to use separate modules. This chapter mainly focuses on these separate ZigBee and WiFi modules, their development environments and evaluation kits.

4.1 ZigBee Development Environment

IEEE 802.15.4, ZigBee and RF4CE compatible transceiver part of the system is based on Texas Instruments CC2530 SoC. TI CC2530 SoC has its own RF transceiver with an industry-standard 8051 MCU, in-system programmable flash memory and 8-KB RAM in the same die for RF communication and hosting the required software stack for ZigBee. In Table 4.1, CC2530 SoC specifications are listed in detail.

Table 4.1: CC2530 SoC system specifications.

System Parameter	Type / Value
MCU	8051
MCU Code Flash size	32/64/128/256 KB
MCU RAM size	8 KB
Antenna Connection	Differential
Data Rate (Max)	250 kbps
Frequency (Max)	2507 MHz
Frequency (Min)	2394 MHz
Operating Voltage	3.3 V
Sensitivity	-97 dBm
TX Power	4.5 dBm

For ZigBee network development and implementation part of the study, TI CC2530 development kit is used. This kit contains 2 SmartRF05 evaluation boards, 5 SmartRF05 battery boards and 7 CC2530 evaluation modules. The boards are easy to plug one to another, contains some jumper and switch options to adapt or test your software without spending to much time on developing hardware. In Figure 4.1, TI CC2530 development kit is shown.

Figure 4.1: TI SmartRF05 Development Kit.

4.1.1 TI CC2530 evaluation module schematic review

To be able to use CC2530 SoC, first we need to understand and analyze the architecture of evaluation module. On CC2530EM, whole system is populated on a small double layer PCB. The module has 2 SMD connectors to break SoC pins out to the other SmartRF05 boards. Since CC2530 SoC has standard UART and SPI peripherals, it provides alternatives according to the needs of communication with other boards. Figure 4.2 and 4.3 shows CC2530EM block diagram and module itself, respectively.

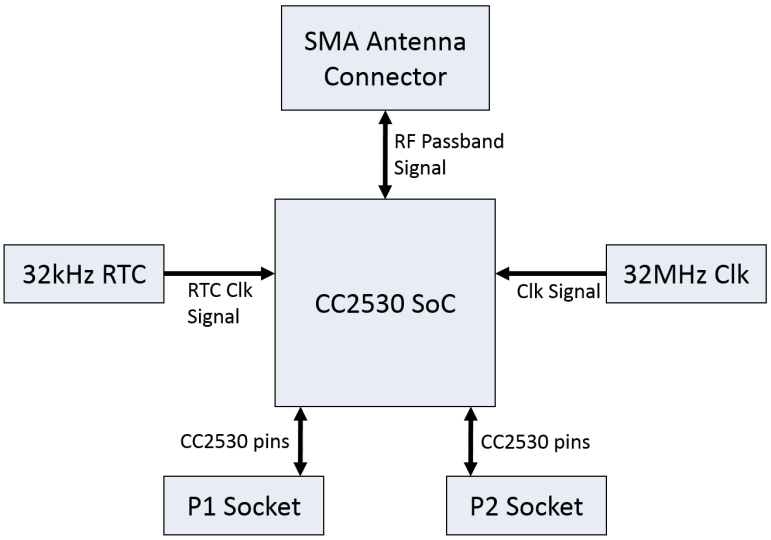


Figure 4.2: CC2530EM block diagram.

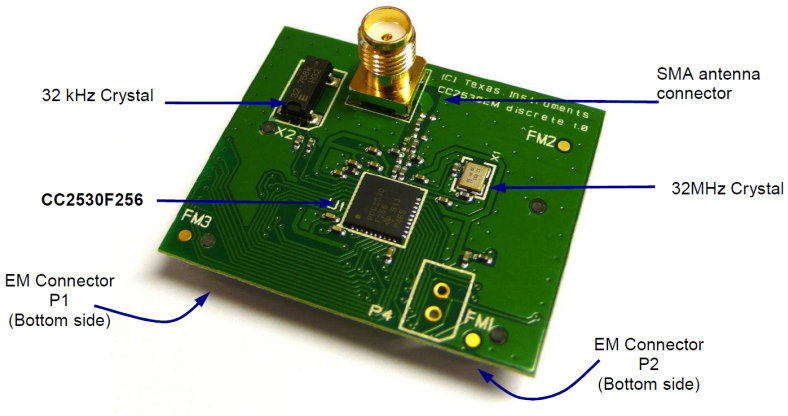


Figure 4.3: CC2530 evaluation module.

Since SPI transport is more suitable for both CC2530-ZNP and WICED module, we will use SPI rather than UART or USB interface.

Following standard SPI signals are used [35]:

- **CLK:** Serial Clock
- **SS:** Slave Select
- **MO:** Master-output slave-input data
- **MI:** Master-input slave-output data

Following signals are required for SPI transaction handling and power management of ZNP [35]:

- **MRDY:** Master ready, active low signal. This signal is set by the application processor when it has data ready to send to the CC2530.
- **SRDY:** Slave ready, bi-modal signal. This signal is set by the CC2530 when it is ready to receive or send data.

To be able to configure CC2530-ZNP for appropriate interface option two additional pins are used [35]:

- **CFG0:** This pin is used to indicate the presence (high) or absence (low) of the 32kHz crystal connected to the CC2530-ZNP. This is the sleep crystal that is used to maintain accurate timing when the device is in sleep mode.
- **CFG1:** This pin is used to indicate transport interface of ZNP. If CFG1 pin is high, the CC2530-ZNP will use the SPI transport mode. Otherwise, it will use the UART transport mode.

Using CC2530-ZNP datasheet [35] and CC2530 ZigBee development kit user's guide [36], pin connections and matching signals of CC2530 SoC are listed in Table 4.2.

Table 4.2: CC2530 pin mapping for the EM and EB connectors.

Signal Name	CC2530 pin	P1 Socket (EM) P5 Header (EB)	P2 Socket (EM) P6 Header (EB)
GND	GND	19 GND	
SS	P1_4	14 EM_CS	
CLK	P1_5	16 EM_SCLK	
MO	P1_6	18 EM_MOSI	
MI	P1_7	20 EM_MISO	
MRDY	P0_3	9 EM_UART_TX	
SRDY	P0_4	3 EM_UART_CTS	
RESET	RESET_N		15 EM_RESET
CFG0	P1_2		17 EM_LCD_CS
CFG1	P2_0		19 EM_JOY_MOVE

4.1.2 TI SmartRF05 evaluation board schematic review

As it is mentioned before, CC2530 EM must be connected to other boards via P1 and P2 sockets to be able to break out SoC pins. One of the options for using CC2530 EM is connecting it to SmartRF05 EB. SmartRF05 EB is a kind of motherboard and it provides not only USB programming interface but also breakout pins for the EM. In Figure 4.4, SmartRF05 evaluation board is shown.

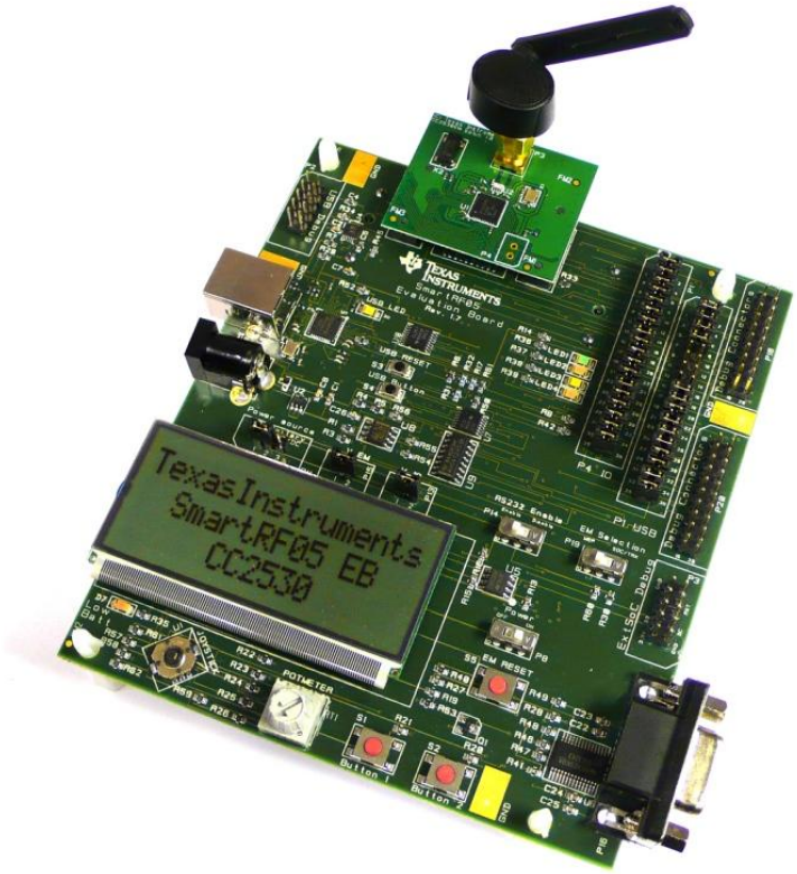


Figure 4.4: SmartRF05 evaluation board.

CC2530 EM's P1 and P2 sockets are connected to the P5 and P6 headers of SmartRF05 EB, respectively. Once we connect the EM board to the EB, CC2530 SoC pins are now broken out to the P18 and P20 pinrows of the SmartRF05 EB. Then, we can easily connect a cable connector to P18 and P19 pinrows for interfacing CC2530. Table 4.3 lists signal and pin mapping for P5, P6 headers and P18,P20 pinrows. Note that

related signals are CC2530-ZNP SPI, interface configuration and master/slave ready indications signals.

Table 4.3: SmartRF05 Evaluation Board pin mapping.

Signal Name	P5 Header	P6 Header	P18 Pinrow	P20 Pinrow
GND	19 GND		20	20
SS	14 EM_CS		14	
CLK	16 EM_SCLK		16	
MO	18 EM_MOSI		18	
MI	20 EM_MISO		12	
MRDY	9 EM_UART_TX		11	
SRDY	3 EM_UART_CTS		13	
RESET		15 EM_RESET		14
CFG0		17 EM_LCD_CS		18
CFG1		19 EM_JOY_MOVE		19

4.1.3 TI CC2530 battery board schematic review

Another alternative for using CC2530 EM is connecting it to the standalone SmartRF05 battery board. This board is a simpler and smaller alternative when compared to the SmartRF05EB. In Figure 4.5, SmartRF05 battery board is shown.

Battery board and EM forms a self power standalone ZigBee node. Similarly, P4, P5 pinrows on SmartRF05BB break out P1 and P2 headers which are directly connected to the EM. In Table 4.4, SmartRF05BB break out pins are listed with ZNP matching signals.

Table 4.4: SmartRF05 Battery Board pin mapping.

Signal Name	P1 Header	P2 Header	P4 Pinrow	P5 Pinrow
GND	20 GND	20 GND	20	20
SS	14 EM_P1_14		14	
CLK	16 EM_P1_16		16	
MO	18 EM_P1_18		18	
MI	20 EM_P1_20		12	
MRDY	9 EM_P1_9		11	
SRDY	3 EM_P1_3		13	
RESET		15 EM_P2_15		14
CFG0		17 EM_P2_17		18
CFG1		19 EM_P2_19		19

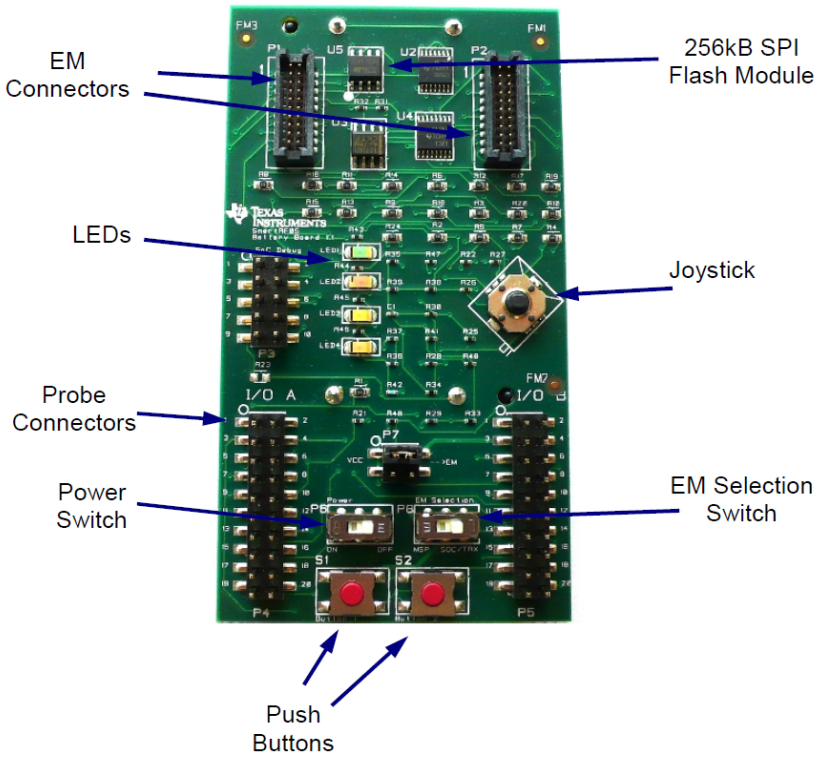


Figure 4.5: SmartRF05 battery board.

4.2 WICED Development Environment

IEEE 802.11n WiFi certified transceiver part of the sytems is based on Broadcom Wireless Internet Connectivity for Embedded Devices platform. WICED platform is built from BCM43362 WLAN SiP and ARM Cortex-M3 based 32-bit ST microcontroller. BCM43362 is called SiP; since all the physical layer baseband and passband modification is done on the same package. In Figure 4.6, Broadcom WICED development board is shown. BCM9WCD1EVAL1 development board comes with BCM943362WCD4 evaluation module on it. There are two ways to program ARM Cortex-M3 uC on WICED module, one of them is direct JTAG interface and the other is via USB to JTAG converter on board. Mini USB port on the board connects to the USB hub controller first. This USB hub provides both JTAG communication and USB to UART debug port connection of the uC at the same time. In this study, mini USB port will be used to program and debug the board, direct JTAG port is ignored.

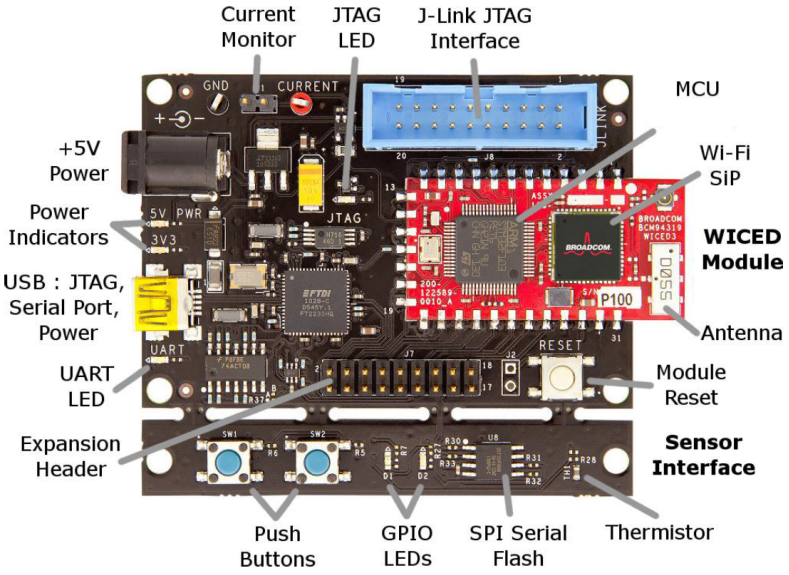


Figure 4.6: Broadcom BCM9WCD1EVAL1 development board.

4.2.1 WICED evaluation board schematic review

Table 4.5: WICED Evaluation Board pin mapping.

Signal Name	STM32 Port	J7 Expansion Header	Board Connection
GND		2 / 17 GND	
MICRO_SPI_SSN	A 4	16 SPI_SS_L	SPI Flash
MICRO_SPI_SCK	A 5	15 SPI_SCK	SPI Flash
MICRO_SPI_MOSI	A 7	13 SPI_MOSI	SPI Flash
MICRO_SPI_MISO	A 6	14 SPI_MISO	SPI Flash
MICRO_GPIO_1	B 7	7 DAC2	LED D2
MICRO_ADC_IN3	A 3	5 ADC3	Thermistor
MICRO_GPIO_0	B 6	6 DAC1	LED D1
MICRO_ADC_IN1	A 1	3 ADC1	Button SW2
MICRO_ADC_IN2	A 2	4 ADC2	Button SW1

4.3 WiFi-ZigBee Embedded IP Gateway Node Hardware

WSN applications generally exploit from low throughput Low Rate-Wireless Personal Area Network (LR-WPAN) protocols. One of the most widespread protocols that uses IEEE 802.15.4 LR-WPAN is ZigBee. Mesh network capability, low-power consumption and open source availability makes ZigBee preferable to other technologies. However, ZigBee protocol, as other LR-WPAN protocols, needs a gateway device to be able to communicate with the devices in IP domain. Thanks to the low cost high performance embedded microcontrollers, implementing embedded IP gateway for a ZigBee network becomes an attainable solution. As one alternative

of quiet a lot solutions, we preferred Broadcom's brand-new wireless Internet connectivity for embedded devices (WICED) platform in combination with TI's CC2530 ZigBee network processor (ZNP) to realize LR-WPAN embedded IP gateway. In this combination, CC2530 has a low power 8051 core while WICED module has ARM Cortex-M3 microprocessor. Since, 8051 core in CC2530 with limited processing power, RAM and ROM size won't be suitable for being the main processor of gateway node, ARM Cortex-M3 core in WICED System in Package (SiP) is chosen as the central processor.

WICED platform comes with two alternatives for IP stack and real time operating system (RTOS): NetX or LwIP TCP/IP stacks and ThreadX or FreeRTOS embedded operating systems. In our application, ARM Cortex-M3 reduced instruction set computing (RISC) microprocessor using ThreadX or FreeRTOS @120MHz on WICED is more suitable for handling IP network interfacing and forwarding tasks of the gateway node, while ZNP (ZigBee Network Processor) is in charge of interfacing ZigBee network. Proposed system architecture can be seen in Figure 4.7. At the right hand side, ZNP system architecure is shown. It uses IEEE 802.15.4 PHY and medium access control (MAC) layers under ZigBee network and application layers. ZNP has a serial peripheral interface (SPI) to communicate with the host processor. At the left hand side, WICED system is shown. WICED features IEEE 802.11n PHY and MAC layers under transport and network layers. Over the transport layer, WebSocket protocol operates as one of the low overhead, full-duplex communication protocols over a single TCP connection.

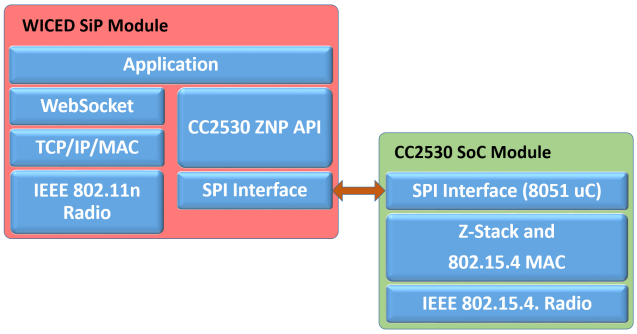


Figure 4.7: LR-WPAN embedded IP gateway system architecture: WICED host MCU and CC2530-ZNP combination.

As soon as the logical connections of the gateway node is achieved, physical connections of the hardware blocks become more and more clear. In Figure 4.8, detailed hardware connectivity of WICED and CC2530 is shown.

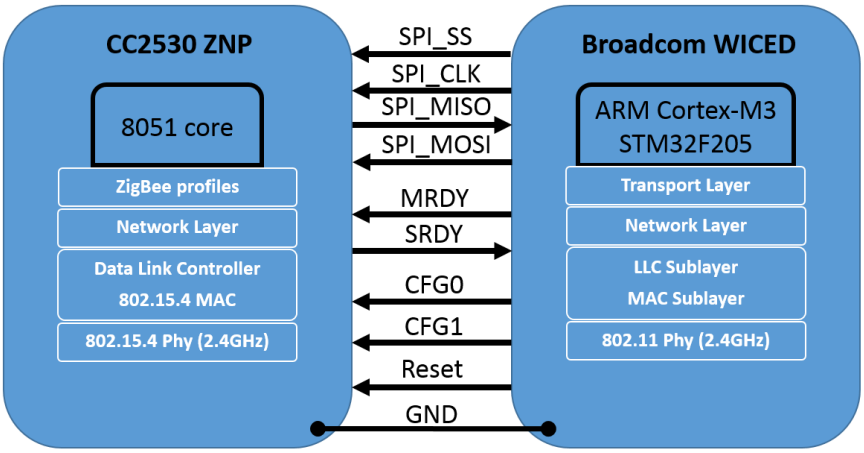


Figure 4.8: Embedded IP gateway node module connections.

Also in Figure 4.9, our WiFi-ZigBee gateway node is shown.

When we think about two wireless technologies at the same node, we have to consider coexistence analysis. IEEE 802.15.4 Sub-1 GHz ZigBee modules combined with a 2.4 GHz or 5 GHz WiFi module doesn't have any coexistence issues. The problem occurs when both of them use the same 2.4 GHz band. If we think about using 2.4 GHz band for both ZigBee and WiFi, we have to consider co-channel interference which will increase PER and retransmissions. In [37] [38] [39], researchers analyzed coexistence of WiFi and ZigBee technologies by applying several coexistence scenarios in a home environment.

4.3.1 WICED, SmartRF05-EB/BB hardware modifications and cabling

WICED WiFi and SmartRF05-BB need some hardware modifications since the modules are connected to several peripheral devices on the boards. Table 4.6 lists the components to be removed.

After completing the hardware modifications and jumper removal operations, we can prepare cables for WiFi and ZigBee board connections. Table 4.7 shows the cable pinnings for both WICED - SmartRF05EB and WICED - SmartRF05BB connections.

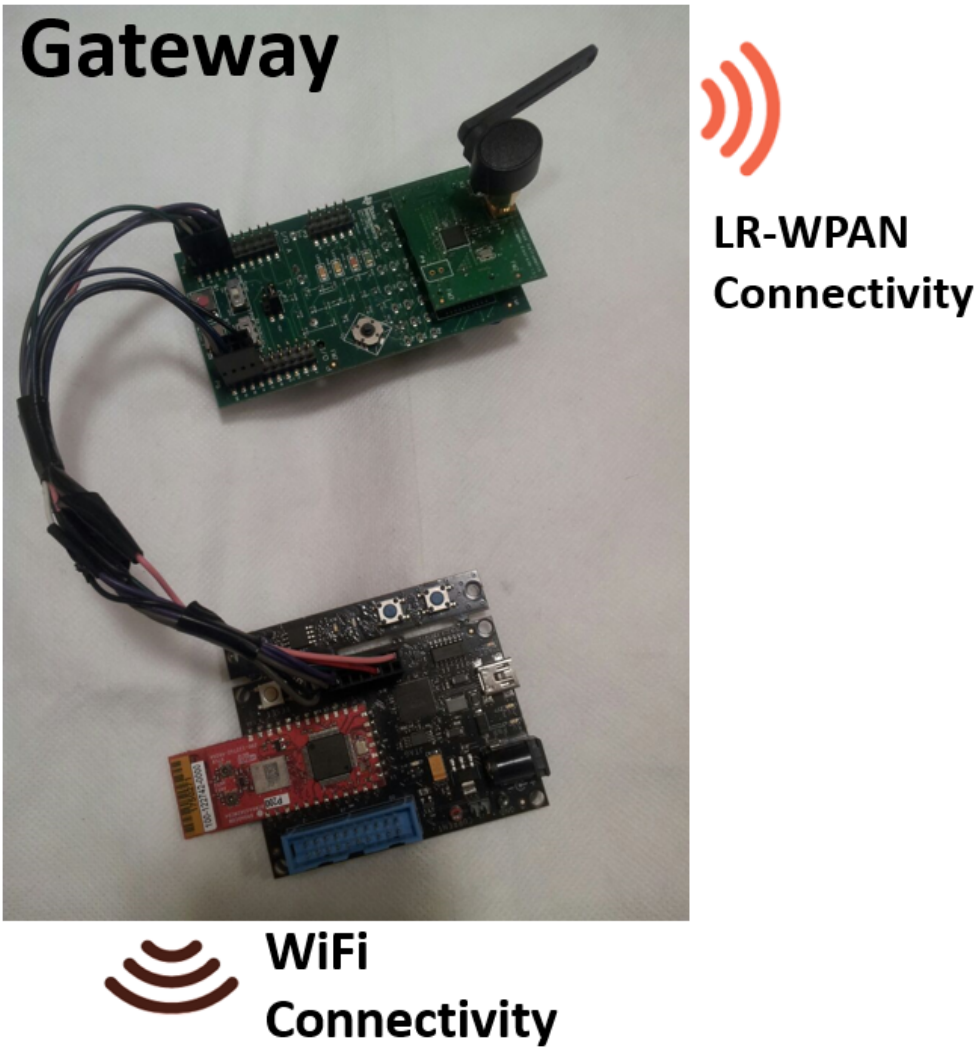


Figure 4.9: WICED WiFi and CC2530 ZigBee embedded gateway node.

Table 4.6: WICED and SmartRF05-EB/BB hardware removals.

WICED Component	SmartRF05-BB Component	SmartRF05-EB Jumper	SmartRF05-EB Jumper
R5	R7	P1 7-8	P10 1-2
R6	R14	P1 9-10	P10 7-8
R7	R15	P1 11-12	P10 9-10
R27	R16	P1 13-14	P10 11-12
R28		P1 15-16	P10 13-14
R30		P1 23-24	P10 29-30
R31		P1 25-26	P10 35-36
R32		P1 27-28	
R33		P1 29-30	
		P1 31-32	
		P1 35-36	

Table 4.7: WICED and SmartRF05-EB/BB pin mapping.

Signal Name	P18 Pinrow (EB) P4 Pinrow (BB)	P20 Pinrow (EB) P5 Pinrow (BB)	J7 Expansion Header
GND	20 GND		2 / 17 GND
SS	14 EM_CS		16 SPI_SS_L
CLK	16 EM_SCLK		15 SPI_SCK
MO	18 EM_MOSI		13 SPI_MOSI
MI	12 EM_MISO		14 SPI_MISO
MRDY	11 EM_UART_TX		7 DAC2
SRDY	13 EM_UART_CTS		3 ADC1
RESET		14 EM_RESET	6 DAC1

4.4 Conclusion of Hardware Developments

The hardware development chapter is an essential part of the thesis work. Since there is a testbed implementation in the scope of the study, hardware development constitutes an important part of it. In this chapter, TI CC2530 ZigBee and Broadcom WICED development environments are analyzed in the scope of hardware. Hardware development kits and the required hardware modifications of their boards are explained in detail for implementing LR-WPAN IP gateway, sensor and actuator nodes. At the end of the studies explained in this chapter, software development stage of the thesis work started.

5. SOFTWARE DEVELOPMENT

The most essential part of the project is software development phase and significant effort made in this period. Due to the fact that different hardware and software platforms must be used for WiFi-ZigBee HetNet implementation, learning process and literature search of each platform take quiet a few time. Thanks to the well documented WICED and TI CC2530-ZigBee development environments, it becomes clear where to start and how to proceed.

First of all, the design requirement for WiFi-ZigBee heterogeneous node was clearly defined. One embedded microprocessor and two radio interfaces are suitable for realizing HetNet node and several HetNet nodes can form an heterogeneous network. In Figure 5.1, HetNet with six double-interface node is shown. It is seen that, this type of node architecture is highly applicable with one WICED WiFi SiP and one CC2530 SoC.

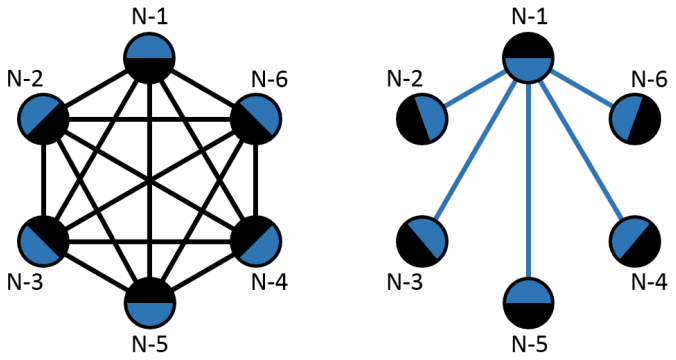


Figure 5.1: ZigBee mesh and WiFi star topology at the same time.

To be able to achieve centralized management of the node interfaces and main application, one of the modules must be used in master and the other in slave configuration. Since, 8051 core in CC2530 with limited processing power, RAM and ROM size won't be suitable for being the main processor of WiFi-ZigBee HetNet node, ARM Cortex-M3 core in WICED SiP is chosen as the central processor. This core is more suitable for handling complex tasks of the HetNet node while ZNP is in charge of interfacing ZigBee network. Proposed system architecture can be seen in Figure 5.2

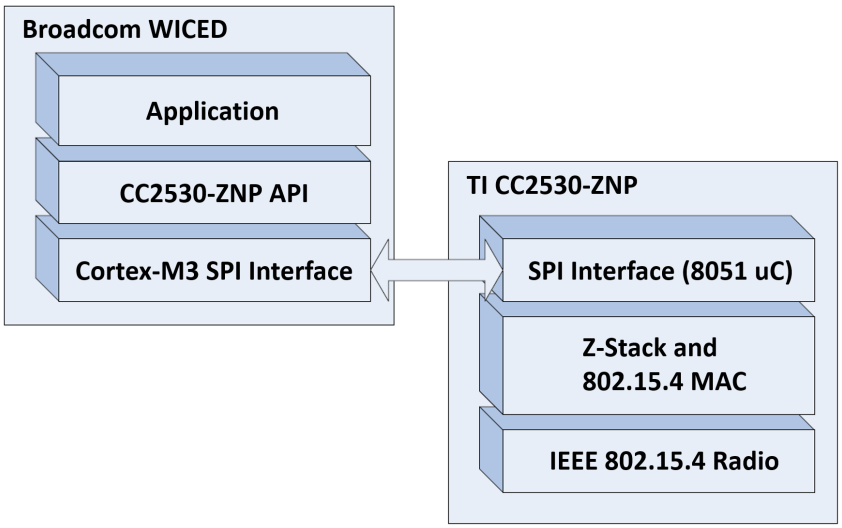


Figure 5.2: WICED host MCU and CC2530-ZNP.

5.1 Software Development using Z-Stack on CC2530 SoC

The prerequisite of software development on CC2530 SoC is having a computer which has Windows XP or later Microsoft Windows operating system. Also .NET 1.1 or later framework must be installed before the installation of Z-Stack package. Z-Stack-Home can be downloaded from TI software development website [40]. myTI account is needed to be able to download packages, it is free to sign-up.

Z-Stack ZNP project files in Z-Stack-Home package is compatible with IAR Embedded Workbench (EW8051) suite of software development tools. These software tools provide all compiling, assembling, linking, downloading and debugging processes of Z-Stack on CC2530 SoC module by using SmartRF05 evaluation board. IAR Embedded Workbench (EW8051) can be downloaded from IAR website [41]. IAR-EW8051 is not a freeware software but time-limited trial licence is available through registration.

5.2 WICED Software Development Kit

Broadcom WICED platform is developed for low cost, low CPU power embedded systems which needs internet connectivity via existing WLANs. It comes with IEEE 802.11n capable embedded WLAN chip, supported by an ARM Cortex-M3 32-bit

RISC uC. Cortex-M3 core rises up to 150 DMIPS @ 120 MHz in WICED SiP, enabling not only WLAN connectivity but also application related processing at the same time. For the time critical operation of the WLAN radio and other application staff, WICED is supported by FreeRTOS or ThreadX alternatively. For the WLAN and upper layer connectivity, it is equipped with LwIP or NetX TCP/IP stacks. The software stack of whole WICED platform is shown in Figure 5.3 briefly. Also, developed software modules within the scope of this thesis study are marked.

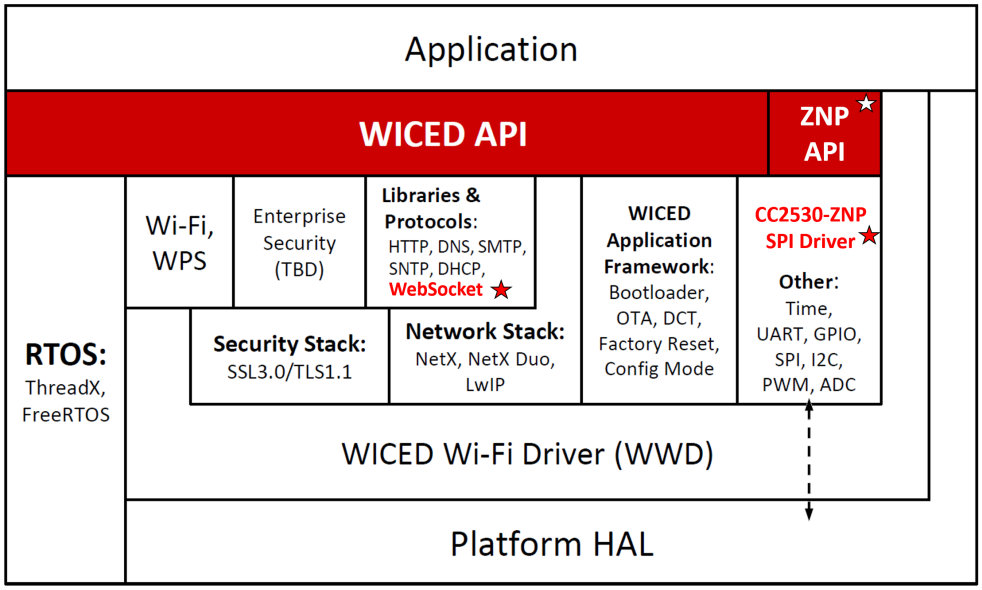


Figure 5.3: WICED software stack. Added protocols and APIs within this thesis study marked with ⋆ (star) sign.

WICED SDK is an Eclipse Helios based software development environment. There are several WICED platforms supported by WICED SDK and one of them is BCM943362WCD4, which is investigated in hardware development chapter in detail. To be able to program and debug the module, mini USB port is used. Figure 5.4 summarizes program and debug alternatives of the WICED board, we will use 1.

In WICED SDK, there are several APIs such as Platform, RTOS, IP communication, WiFi 802.11 etc. These APIs enables the programmer to access libraries, protocols and hardware level controls in WICED system through the development process.

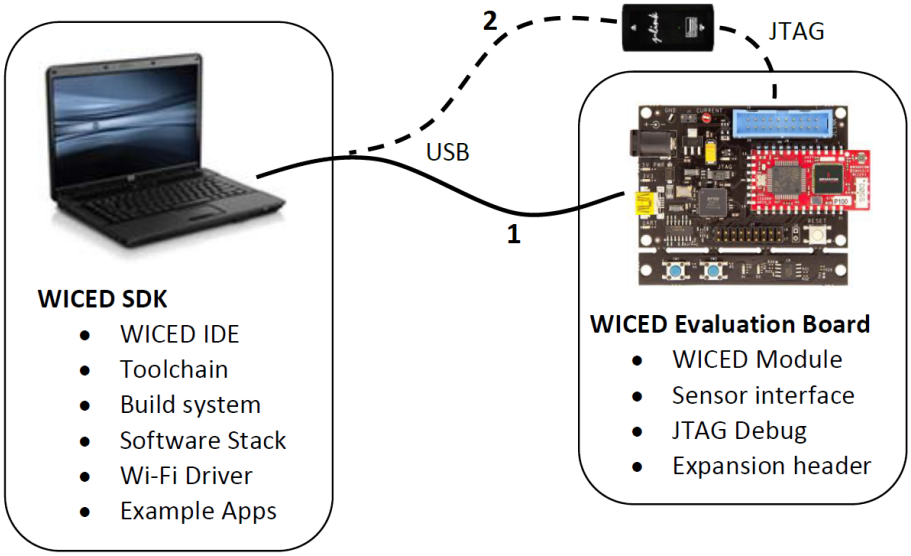


Figure 5.4: WICED development system is depicted. WICED software development kit is on the left hand side and WICED evaluation board is on the right hand side.

5.3 WiFi-ZigBee IP Gateway Software Architecture

5.3.1 ZigBee Network Processor driver software

In Z-Stack package, ZNP project is available for software development purposes. In ZNP solution, the Z-Stack which is compatible with ZigBee Pro, runs on a CC2530 ZigBee SoC. The CC2530-ZNP handles all the ZigBee protocol and communication tasks, thus external microcontroller can be used for different purposes. There are 3 options for communicating with CC2530-ZNP: SPI, UART or USB interface. SPI is more suitable for our application since WICED has also SPI option and SPI has higher data throughput than UART. CC2530-ZNP supported SPI configuration is as follows.

- SPI Slave
- Clock speed up to 4 MHz
- Clock polarity 0 and clock phase 0
- Bit order MSB first

Up to 4 MHz SPI clock speed support of ZNP enables us to achieve even the maximum data rate of 250 kbps which is ZigBee protocol limit specified in CC2530 specifications Table 4.1 in Section 4.1.

Before we start to implement ZNP driver software, we need to understand signal operations of CC2530-ZNP. First of all, the configuration pins must be set to appropriate level. High level on CFG0 pin indicates that there is an external RTC clock. Since CC2530EM has this external crystal clock populated on the board, CFG0 pin must be set to high. Then, CFG1 pin must be set to high level, because we use SPI transport mode on ZNP. These two pins are set to high level by using pull-up resistors on the cable between WICED and CC2530-ZNP which is explained in Section 4.3.1. It was also possible to connect WICED GPIOs to set CFG0 and CFG1 signals but it is not necessary to change the signals frequently. After setting the configuration pins, we are now ready to dive into the ZNP interface signal operations.

First we configure SPI peripheral of WICED for communicating with CC2530-ZNP. Maximum SPI speed of 4 MHz is used with clock polarity 0, clock phase 0 and MSB first on data lines. DMA mode is also activated for the SPI interface to use dedicated DMA unit in ARM Cortex-M3 uC. With using DMA, CPU is not fully occupied for the entire duration of the read/write cycle but it only starts and controls the process. Following code block performs SPI peripheral initialization operation in WICED API structure.

```
static wiced_spi_device_t spi_ZNP = {
  .port = WICED_SPI_1,
  .chip_select = WICED_SPI_FLASH_CS,
  .speed = 4000000,
  .mode = ( SPI_CLOCK_RISING_EDGE | SPI_CLOCK_IDLE_LOW |
            SPI_USE_DMA | SPI_MSB_FIRST ),
  .bits = 8 };
wiced_spi_init( &spi_ZNP );
```

After initializing SPI peripheral, we can analyze ZNP SPI transactions. ZNP SPI transport signals operate according to the following rules:

- The application processor initiates a transaction by setting MRDY pin low and then waits for SRDY pin to go low.

- The application processor shall never set MRDY pin high to end a transaction before all bytes of the frame have been transferred.
- When receiving a POLL or SREQ command, the CC2530 shall set SRDY pin high when it has data ready for the application processor.
- When receiving an AREQ command, the CC2530 shall set SRDY pin high when all bytes of the frame have been received.

There are 3 commands in CC2530-ZNP SPI transport protocol: AREQ, POLL and SREQ:

AREQ command: AREQ stands for Asynchronous Request and it can be sent either from the application processor to the CC2530-ZNP or from the CC2530-ZNP to the application processor. Figure 5.5 shows AREQ command sent from application processor to ZNP.

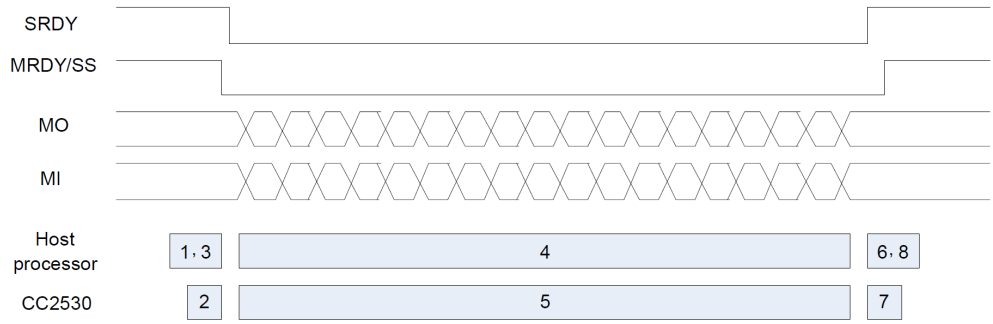


Figure 5.5: CC2530-ZNP AREQ command.

It is possible to track AREQ command signal timings from Figure 5.5. The following sequence of events occurs in case of an AREQ command:

1. Application processor has an AREQ frame to send. It sets MRDY pin low and waits for SRDY pin to go low.
2. CC2530-ZNP receives falling edge of MRDY. When ready to receive data it sets SRDY pin low.
3. Application processor reads SRDY pin low. It starts data transmission.
4. Application processor transmits data until frame is complete.

5. CC2530 receives data until frame is complete.
6. Application processor waits for SRDY pin to go high.
7. CC2530 receives complete frame and sets SRDY pin high.
8. Application processor reads SRDY pin high. It sets MRDY pin high.

POLL command: POLL stands for Polling and it is sent from the application processor to the CC2530-ZNP. Figure 5.6 shows POLL command.

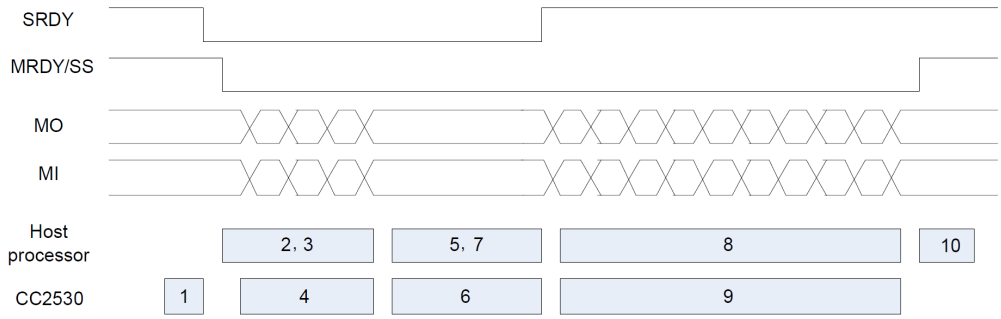


Figure 5.6: CC2530-ZNP POLL command.

It is possible to track POLL command signal timings from Figure 5.6. The following sequence of events occurs in case of a POLL command:

1. CC2530 has an AREQ frame to send. When ready to receive data set SRDY pin low.
2. Application processor detects SRDY pin low and sets MRDY pin low. It prepares POLL command and start data transmission.
3. Application processor transmits data until frame is complete.
4. CC2530 receives data until frame is complete.
5. Application processor waits for SRDY pin to go high.
6. CC2530 prepares AREQ frame for transmission. When ready to transmit set SRDY pin high.
7. Application processor reads SRDY pin high. Start data reception.
8. Application processor receives data until frame is complete.
9. CC2530 transmits data until frame is complete.

10. Application processor receives complete frame. Set MRDY pin high.

SREQ command: SREQ stands for Synchronous Request and it is sent from the application processor to the CC2530-ZNP. As an answer from CC2530-ZNP to application processor, SRSP (Synchronous Response) frame is sent. Figure 5.7 shows POLL command.

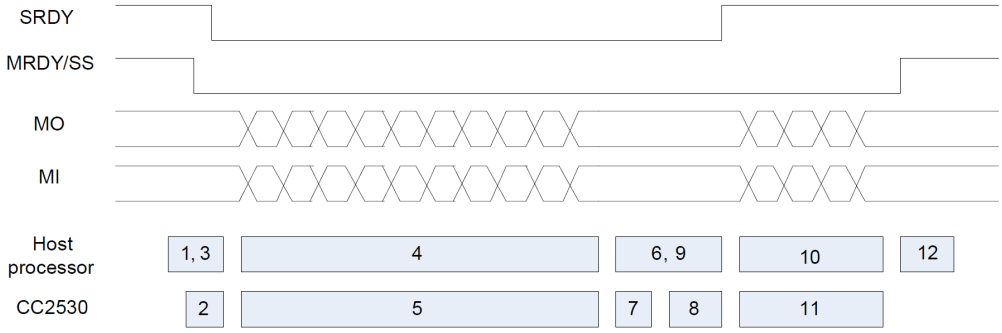


Figure 5.7: CC2530-ZNP SREQ command and SRSP response.

It is possible to track SREQ command signal timings from Figure 5.7. The following sequence of events occurs in case of a SREQ command:

1. Application processor has an SREQ frame to send. Set MRDY pin low and wait for SRDY pin to go low.
2. CC2530 receives falling edge of MRDY. When ready to receive data set SRDY pin low.
3. Application processor reads SRDY pin low. Start data transmission.
4. Application processor transmits data until frame is complete.
5. CC2530 receives data until frame is complete.
6. Application processor waits for SRDY pin to go high.
7. CC2530 processes SREQ command and executes function.
8. CC2530 prepares SRSP frame. When ready to transmit data set SRDY pin high.
9. Application processor reads SRDY pin high. Start data reception.
10. Application processor receives data until frame is complete.
11. CC2530 transmits data until frame is complete.

12. Application processor receives complete frame. Set MRDY pin high.

As we can understand from command transport events between application processor and ZNP, there are 2 important signals called MRDY and SRDY. SRDY signal is controlled by CC2530-ZNP and it is time critical asynchronous signal. We need to design our driver algorithm according to this limitation. The most suitable way to handle this SRDY signal is using GPIO interrupt function of WICED MCU. In this way we can give response to ZNP as soon as possible. By the way, we mustn't affect RTOS's usual processing operation by expanding interrupt handler function. To prevent this, we generally use interrupt service routines only for pushing asynchronous event to the related RTOS thread. This maintains usual RTOS operation while exploiting from asynchronous interrupts.

The code block in Appendix 1.2 enables both edge interrupt request for SRDY signal, creates RTOS thread for interrupt event and constitutes relation between *SRDY Handler Event* function and RTOS thread.

Now we are ready to respond to CC2530-ZNP SRDY signal immediately, but how can we relate SRDY signal with the interface states between WICED module and CC2530-ZNP. What is the efficient way of doing this? Finite State Machine software model helps us to achieve this. If we are able to establish an appropriate FSM model, we can realize it using Switch-Case structure in C programming language [34] [42]. Following two interoperating FSM models in Figure 5.8 meet the needs of SPI transport protocol between WICED module and CC2530-ZNP. These two FSMs work at the same time by using RTOS threads and scheduling functionality which is mentioned in Chapter 2. The algorithm and C code that realizes both interrupt handler and application FSM functionality for SRDY signal triggers from CC2530-ZNP can be found in Appendix 1.3.

Up to this point, we have implemented FSM based real time ZNP SPI transport handler functions by exploiting GPIO interrupt and RTOS event scheduler functionalities of ARM-Cortex M3 core on WICED module. The last but not the least, real time input/output buffering of data packets are also very critical operation for this type of systems because time-critical processing and forwarding of data packets are important. The system cannot process incoming data immediately after a SRDY interrupt event,

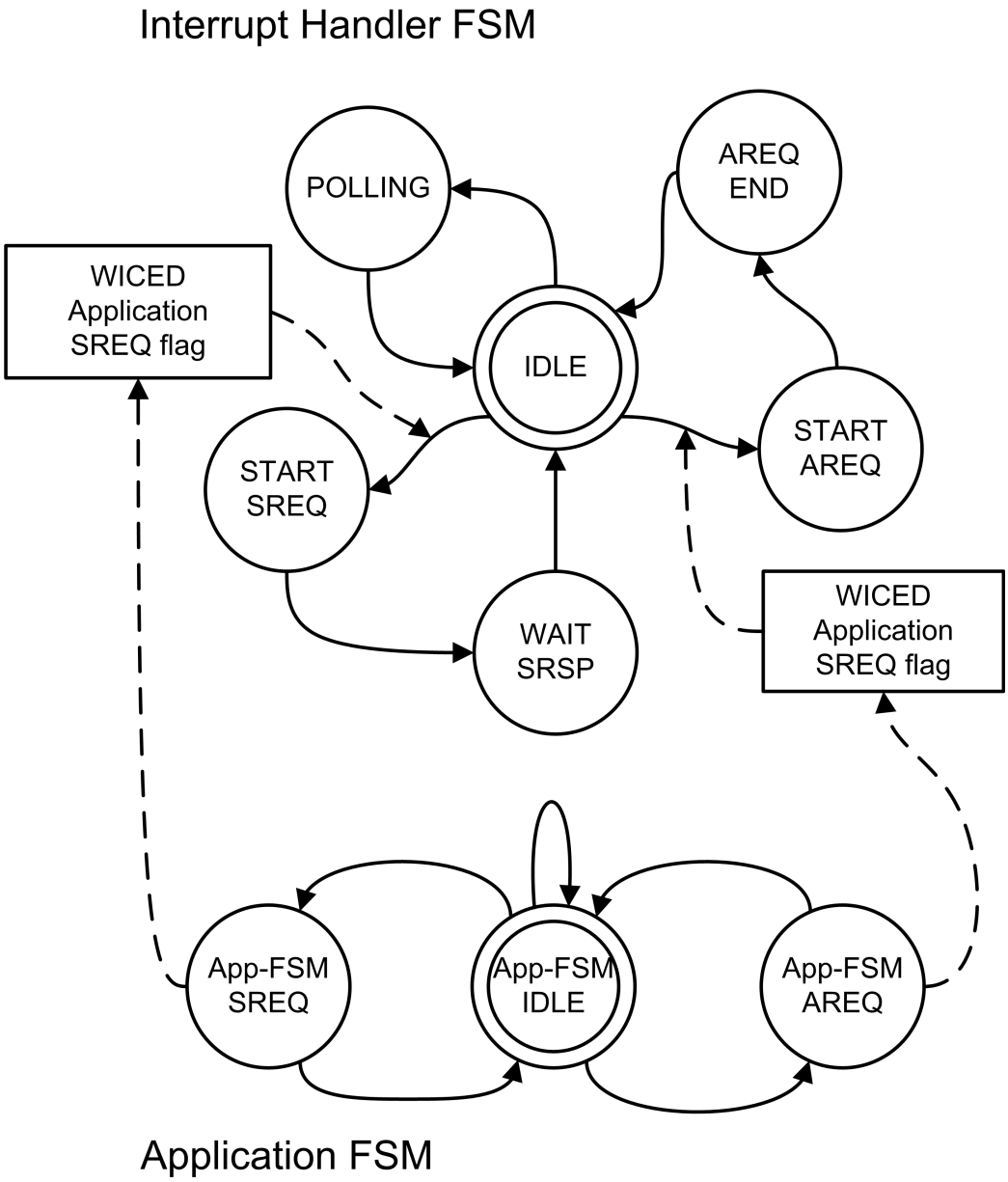


Figure 5.8: SRDY interrupt handler FSM and application FSM state diagrams and their relations.

because dedicating processor time to a single task for long periods may hinder response to other events. To prevent this in an efficient way, circular queue structure can be used [42] [34]. To understand what type of data packets will be transferred between WICED and ZNP, look at the Figure 5.9. In this figure, ZNP general frame format is shown.

Note that first byte of the frame defines the length of the payload in the packet. Since variable size data items can be transferred, length information of the payload must be

Bytes: 1	2	0-250
Data Length	Command	Data

Figure 5.9: CC2530-ZNP general frame format.

used for enqueue and dequeue operations. It is now clear that, variable size data FIFO buffer is suitable for our application. This buffer type will be used both IN and OUT direction for not only payload data but also CC2530-ZNP configuration and response commands. In Figure 5.10, variable size data FIFO circular queue is shown. Note that DeQptr points the first available address for reading operation while EnQptr points the first available address for writing operation. When the number of available registers, namely NBavail, is decreased to zero, buffer overrun will occur, so it is very important to choose a right value for queue size according to the application. The algorithm and C code that realizes variable size FIFO circular queue structure for 8-bit unsigned character data type can be found in Appendix 1.4.

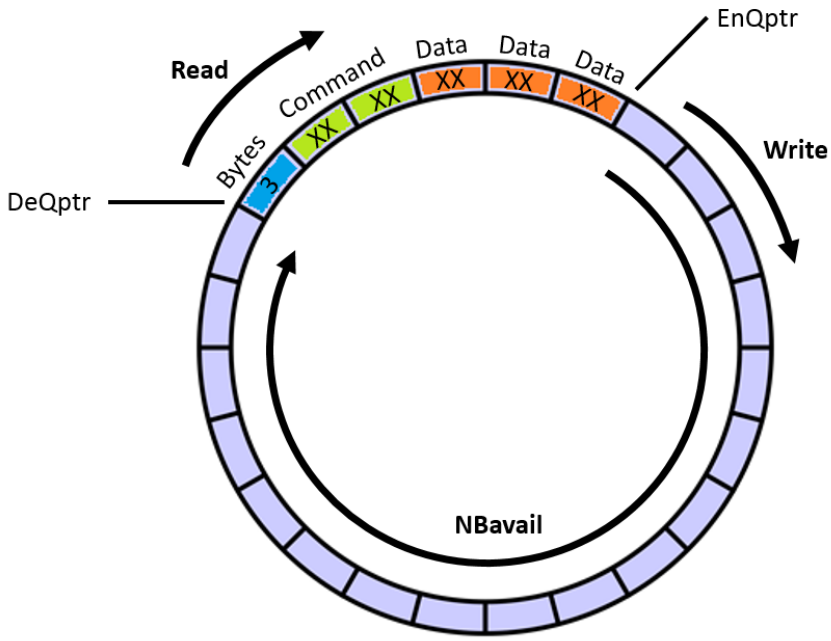


Figure 5.10: Variable size data FIFO circular queue structure.

5.3.2 ZigBee-WiFi IP Gateway XOR Coding Software

The ZigBee-WiFi IP gateway node developed in this study has network coding capability by XORing the packets while transferring between end-devices in Low-Rate

Wireless Personal Area Network (LR-WPAN). The gateway node is a ZigBee coordinator device in the network. Other sensors and actuators are connected to the coordinator and transfers data periodically to the other nodes in the network. Gateway node, namely coordinator collects the packets and XORs them while transferring. In this way, 3 transmission cycles are used for each data sharing cycle instead of 4 cycles. In the long run, this 1 transmission cycle saving provides about 0.25 saving of both TDMA slot and node power consumption.

As it is explained and depicted in Section 1.4, there must be 3 nodes for XOR coding scenario. Consider a scenario with 2 end-devices and 1 coordinator node. End-devices will send their messages to other end-devices through the coordinator node and there will be more than one possible receivers. To be able to realize XOR coding on a ZigBee coordinator node and manage packet forwarding operation, several message headers are defined.

- **AtoB:** Indicates that the packet is uncoded and receiver of the message is Bob
- **BtoA:** Indicates that the packet is uncoded and receiver of the message is Alice
- **SEND:** Synchronization packet sent by the coordinator

At the first stage of the connection, coordinator sends synchronization packet to be able to synchronize the nodes for periodic data transmission. After the synchronization, end-devices may transmit packets to the other nodes in their time slot by specifying the destination node using the header defined above. If there is no two-way data for transferring, coordinator node waits for the timeout and transmits any one-way packet uncoded not to delay the traffic. Figure 5.11 summarized the synchronization and XOR coding scenarios.

The algorithm and C code block of the XOR coding application can be found in Appendix 1.5.

5.4 IoT Overall System Architecture

Up to this section, client device LR-WPAN and gateway WLAN part of the system is implemented. To be able to enable these end devices to send and receive data

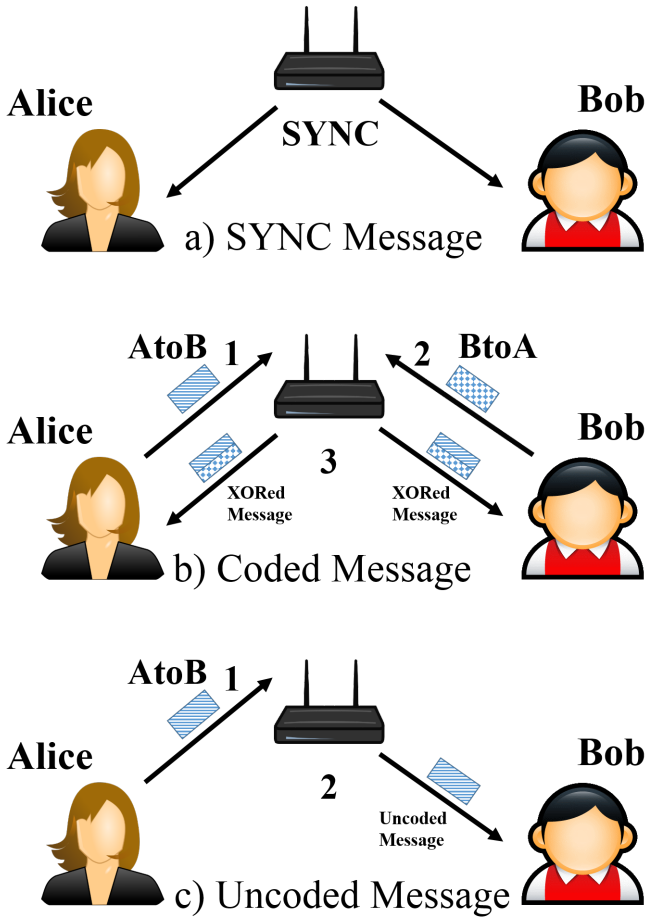


Figure 5.11: XOR coding stages and scenarios. a) Node synchronization frame sent by the coordinator. b) 2-way communication scenario and XOR coded packet transfer. c) 1-way communication scenario and uncoded message transfer after timeout.

over the Internet, we need to define overall system architecture including cloud side. By definition, IoT paves the way for pervasive scaling of Internet enabled devices. Therefore, it should have expandable structure in the aspect of both simultaneously served devices and number of associated services to those devices. Furthermore, service oriented architecture [30] [31] design pattern should be applied for the efficient scaling of the VAS cloud analytics. The architecture that meets all of those needs is defined in Figure 5.12.

On the right hand side of Figure 5.12, IoT device side implementation is represented by a smarthome sensor network. There is a gateway node in between LR-WPAN sensors and IP network infrastructure. Regardless of LR-WPAN and WAN protocols, there must be a kind of packet routing and forwarding mechanism in between. On the left hand side, proposed system for IoT cloud side is shown. There is a database for storing user and device related data and their relation in between. There are more than

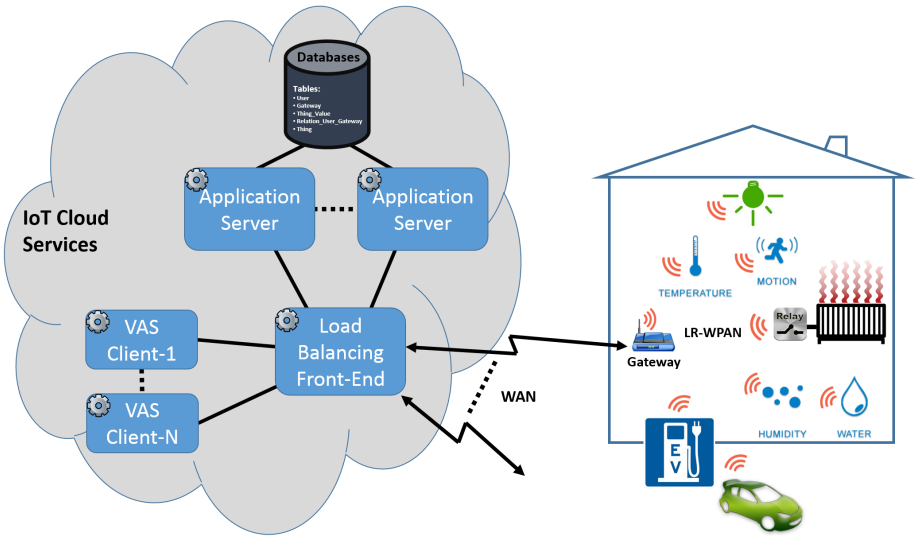


Figure 5.12: Expandable overall system architecture including value-added cloud services.

one application servers and this number can be increased according to the needs and number of services to be supported by the system. Also it is shown in the figure that, there is a load balancing front-end application at the heart of the cloud architecture. The purpose of this load-balancing front end becomes more clear when we think of the number of More detailed database and application model of the cloud services is shown in Figure 5.13.

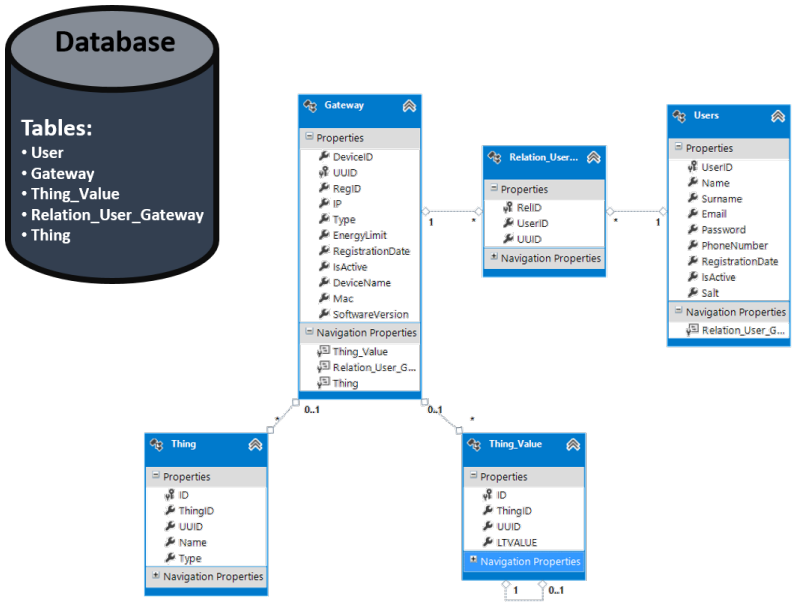


Figure 5.13: Database components and their relations with the other components.

This architecture is implemented just for testing the operation of LR-WPAN network coding part. Server side analysis of the proposed system is not in the scope of this study.

5.5 Conclusion of Software Development

In this chapter, software development stage of the thesis is included. First, software development environments of ZigBee and WICED WiFi are introduced, then software stacks and libraries are revisited. Since WICED WiFi ARM Cortex-M3 is the host processor for this application, ZNP driver software development in WICED elaborately explained by giving layered architecture of WICED-ZNP logical architecture.

6. CONCLUSIONS AND RECOMMENDATIONS

As the Internet has become more and more important in daily life of the society, Internet related applications bloomed and today it is hard to think the society being devoid of the usage and utilization of the Internet. Almost all sectors and industries including telecommunications, state, health, education, banking and payment, transportation, manufacturing, tertiary, sales and marketing utilizes from efficiency, speed, mobility and manageability of the Internet related services and applications. As the communications technologies are evolving from "things on the Internet" concept to the "Internet of Things" concept, the society will need more and more efficient, fast, mobile, easy to manage and secure Internet related services. To achieve this goal, academic researches and industrial applications should be matched for the same objective, empowering each other to reach mutual and common goal. Humbly, our aim is to combine novel efficient and useful approaches in academic researches with novel and available networking technologies in the market. Frankly speaking, our goal of putting a brick for future Internet of Things and networked society is achieved.

One of the novel researches in networking and communication technologies is network coding approach for efficient and reliable data transmission between devices, especially for wireless. In Chapter 1 and Chapter 2, network coding technique is mentioned in detail. Researchers first suggested network coding techniques for the network information transmission in the study [11]. Then, several researches have been conducted in the scope of network coding including the studies [14] [15] [16] [13]. On the other hand, networked society have met a new concept called Internet of Things, which suggests to connect any object in the world to the Internet for collecting and sharing data from any sensor around. As might be expected, this concept requires higher capacity utilization of already deployed internetwork system and efficiency in radio spectrum since almost all of the connected objects will use wireless technologies. For this purpose, we aimed to converge the needs of new internetwork system with the

promising advantages of network coding approaches for device side communication and service oriented design [30] [31] advantages for the cloud services management and cloud data analytics.

At the end of this thesis study, end-to-end IoT system infrastructure is realized using low power wireless CC2530 ZigBee and WICED WiFi modules at the front-end and service oriented cloud management server at the infrastructure back-end. Heterogeneous radio nodes including WiFi and ZigBee radio interfaces are designed for IP gateway functionality of LR-WPAN ZigBee network. XOR coding is implemented and tested for a 3 node LR-WPAN, including 1 sensor, 1 actuator and 1 IP gateway node. Not only software and hardware implementations in wireless personal area devices but also design of overall system infrastructure are completed.

6.1 Practical Applications and Future Work

The Internet of things ecosystem will form the future of networked society. Having easy deployment and serviceability features, wireless sensor and actuator networks will be in the center of this IoT ecosystem. In this manner, the proposed system architecture in this study is highly applicable for smart-home, smart-office, smart-grid and smart-city applications with integrated value added services related to these applications. With the utilization of increased multicast transmit capacity of network coding techniques in LR-WPAN environment and low overhead communication protocols, total number of simultaneously served devices increased because of the fact that the utilization from radio spectrum is scaled up. Also, it is stated and supported by the simulation results in Chapter 2 that reduced retransmissions and increased throughput rates provides low power usage for sensor devices, enabling longer battery life for battery powered sensor and actuator devices.

As being the most widespread standard in the market for sensor and actuator network devices, IEEE 802.15.4 LR-WPAN standard is almost the only exhaustive PHY and MAC standard used for LR-WPAN sensors. There are various upper layer protocols and specifications such as ZigBee, Z-Wave and 6LoWPAN which uses IEEE 802.15.4 PHY and MAC. Since those upper layer protocols are widespread in the market being used for end-user products, the proposed system in this thesis work can easily adapted to the market wide applications.

Several potential future works are predicted according to the needs of IoT infrastructure and concepts suggested in this thesis. One of them is applying network coding techniques for capacity increase and load balancing in cloud services. Since network coding techniques provide increased multicast transmit capacity, application of network coding approach in the design of cloud service network architecture can provide increase in the utilization of cloud server capacity. The other one is using network coding techniques in between PHY and MAC layers instead of upper protocol layers, to be able to utilize from cross layer techniques and improvements, which provides more efficient protocol development for network coding in LR-WPAN. The last but not the least, security and privacy related studies and applications will become more and more important as IoT becomes widespread. Since random network coding techniques make use of several coding algorithms and may exploit from finite field arithmetics as most of the cryptography algorithms do, it can be suggested to combine both network coding and cryptography features in the same coding technique. This will provide both bandwidth efficiency and security in the same coding algorithm.

REFERENCES

[1] **Katti, S.**, **Rahul, H.**, **Hu, W.**, **Katabi, D.**, **Medard, M. and Crowcroft, J.** (2006). XORs in The Air: Practical Wireless Network Coding, *SIGCOMM*, Pisa, Italy.

[2] **Url-3**, `https://ixpotwasher.files.wordpress.com/2014/09/st_julians_congested_24ghz.png`, available on 21.11.2014.

[3] **Dong, Q.**, **Wu, J.**, **Hu, W. and Crowcroft, J.** (2007). Practical Network Coding in Wireless Networks, *ACM MobiCom*, Montréal, Québec, Canada.

[4] **Evans, D.**, (2011), The Internet of Things How the Next Evolution of the Internet Is Changing Everything, Cisco IBSG, White Paper.

[5] (2011), More than 50 Billion Connected Devices, Ericsson, White Paper.

[6] **Stankovic, J.A.** (2014). Research Directions for the Internet of Things, *IEEE Internet of Things Journal*, ***1***(1).

[7] **Vermesan, O. and Friess, P.**, (2013). Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems, River Publishers, Aalborg.

[8] **Hunter, T.E. and Nosratinia, A.** (2002). Cooperation Diversity through coding, *IEEE International Symposium on Information Theory*, Lausanne, Switzerland.

[9] **Chen, Y.**, **Kishore, S. and Li, J.** (2006). Wireless Diversity through Network Coding, *IEEE Wireless Communications and Networking Conference*, Las Vegas, USA.

[10] **Xiao, L.**, **Fuja, T.**, **Kliewer, J. and Jr., D.C.** (2007). A network coding approach to cooperative diversity, *IEEE Transactions on Information Theory*, ***53***(10), 3714–3722.

[11] **Ahlswede, R.**, **Cai, N.**, **Li, S.Y.R. and Yeung, R.W.** (2000). Network information flow, *IEEE Transactions on Information Theory*, ***46***(4), 1204–1216.

[12] **Deb, S.**, **Effros, M.**, **Ho, T.**, **Karger, D.R.**, **Koetter, R.**, **Lun, D.S.**, **Medard, M. and Ratnakar, N.** (2005). Network coding for wireless applications: A brief tutorial, *IWWAN*, London, UK.

[13] **Ho, T.**, **Koetter, R.**, **Medard, M.**, **Karger, D. and Effros, M.** (2003). The Benefits of Coding over Routing in a Randomized Setting, *ISIT*, Yokohama, Japan.

[14] **Ho, T. and Lun, D.S.**, (2008). Network Coding, Cambridge University Press, Cambridge.

[15] **Deb, S.**, **Medard, M. and Choute, C.** (2006). Algebraic gossip: A network coding approach to optimal multiple rumor mongering, *IEEE Transactions on Information Theory*, ***52***(6), 2486–2507.

[16] **Katti, S.**, **Katabi, D.**, **Hu, W.**, **Rahul, H. and Medard, M.** (2005). The importance of being opportunistic: Practical network coding for wireless environments, *43rd Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, USA.

[17] **Koetter, R. and Medard, M.** (2003). An algebraic approach to network coding, *IEEE/ACM Transactions on Networking*, ***11***(5), 782–795.

[18] **Couto, D.S.J.D.**, **Aguayo, D.**, **Bicket, J. and Morris, R.** (2003). A high-throughput path metric for multi-hop wireless routing, *ACM MobiCom*, SanDiego, California, USA.

[19] **Kohler, E.**, **Morris, R.**, **Chen, B.**, **Jannotti, J. and Kaashoek, M.F.** (2000). The click modular router, *ACM Transactions on Computer Systems*, ***18***(3), 263–297.

[20] **Qureshi, J.**, **Foh, C.H. and Cai, J.** (2014). Online XOR packet coding: Efficient single-hop wireless multicasting with low decoding delay, *Computer Communications*, ***39***, 65–77.

[21] **Zhang, S.**, **Liew, S.C. and Lam, P.P.** (2006). Hot topic: physical-layer network coding, *ACM MobiCom*, Los Angeles, USA.

[22] **Rouayheb, S.**, **Chaudhry, M. and Sprintson, A.** (2007). On the Minimum Number of Transmissions in Single-Hop Wireless Coding Networks, *IEEE ITW*, California, USA.

[23] **Chi, K.**, **Jiang, X. and Horiguchi, S.** (2010). Network coding-based reliable multicast in wireless networks, *Computer Networks*, ***52***(11), 1823–1836.

[24] **Rozner, E.**, **Iyer, A.P.**, **Mehta, Y.**, **Qiu, L. and Jafry, M.** (2007). ER: Efficient Retransmission Scheme for Wireless LANs, *ACM CoNEXT*, New York, USA.

[25] **Lu, L.**, **Xiao, M.**, **Skoglund, M.**, **Rasmussen, L.**, **Wu, G. and Li, S.** (2010). Efficient Network Coding for Wireless Broadcasting, *IEEE WCNC*, Sydney, Australia.

[26] **Kwan, H.Y.**, **Shum, K. and Sung, C.W.** (2011). Generation of innovative and sparse encoding vectors for broadcast systems with feedback, *IEEE ISIT*, St. Petersburg, Russia.

[27] **Shojania, H. and Li, B.** (2009). Random network coding on the iPhone: fact or fiction?, *ACM NOSSDAV*, New York, USA.

[28] **Vingelmann, P.**, **Pedersen, M.V.**, **Fitzek, F.H.P. and Heide, J.** (2010). Multimedia Distribution using Network Coding on the iPhone Platform, *ACM MCMC*, Firenze, Italy.

[29] **Ho, T.**, **Médard, M.**, **Koetter, R.**, **Karger, D.R.**, **Effros, M.**, **Shi, J. and Leong, B.** (2006). A Random Linear Network Coding Approach to Multicast, *IEEE Transactions on Information Theory*, ***52***(10), 4413–4430.

[30] **Endrei, M.**, **Ang, J.**, **Arsanjani, A.**, **Chua, S.**, **Comte, P.**, **Krogdahl, P.**, **Luo, M. and Newling, T.** (2004). *Patterns: Service-Oriented Architecture and Web Services*, Redbooks, IBM.

[31] **Lin, G.C.**, **Desmond, K.E.T.**, **Htoon, N.T. and Thuat, N.V.** (2011). *A Fresh Graduate's Guide to Software Development Tools and Technologies, Chapter 10: Service Oriented Architecture*, School of Computing, National University of Singapore.

[32] (2011), Part 15.4: Low-Rate Wireless Personal Area Networks, Computer Society, IEEE.

[33] **Al-Majeed, S.S.**, **Hu, C.L. and Nagamalai, D.** (2011). *Advances in Wireless, Mobile Networks and Applications*, Springer, Heidelberg, Berlin.

[34] **Ozdemirel, B.**, (2011), EE443-Embedded Systems, Lecture Notes, IYTE-EEE.

[35] (2012), ZigBee PRO Network Processor, Texas Instruments, Datasheet.

[36] (2011), CC2530 ZigBee Development Kit User's Guide, Texas Instruments, Datasheet.

[37] **Dominguez, F.**, **Touhafi, A.**, **Tiete, J. and Steenhaut, K.** (2012). Coexistence with WiFi for a Home Automation ZigBee product, *IEEE 19th Symposium on Communications and Vehicular Technology*, Benelux.

[38] **Han, T.**, **Han, B.**, **Zhang, L.**, **Zhang, X. and Yang, D.** (2012). Coexistence study for WiFi and ZigBee under smart home scenarios, *IEEE 3rd International Conference on Network Infrastructure and Digital Content*, Beijing, China.

[39] **Zhang, X. and Shin, K.G.** (2013). Cooperative Carrier Signaling: Harmonizing Coexisting WPAN and WLAN Devices, *IEEE/ACM Transactions on Networking*, ***21***(2), 426–439.

[40] **Url-1**, `http://www.ti.com/tool/z-stack&DCMP=HPA_RFIC_General&HQS=Other+OT+z-stack`, available on 05.10.2014.

[41] **Url-2**, `http://www.iar.com/Products/IAR-Embedded-Workbench/8051/`, available on 05.10.2014.

[42] **Lee, E.A. and Seshia, S.A.** (2011). *Introduction to Embedded Systems - A Cyber-Physical Systems Approach*, LeeSeshia.org.

APPENDICES

APPENDIX A.1 : The algorithm and Matlab code for random linear network coding simulations
APPENDIX A.2 : ZigBee SRDY signal triggered asynchronous event generation C code block
APPENDIX A.3 : ZigBee SRDY signal interrupt handler function C code block
APPENDIX A.4 : Variable data size circular queue alghorithm and C code block for ZigBee interface data traffic
APPENDIX A.5 : XOR coding application algorithm and C code block.
APPENDIX A.6 : Publication on the thesis: "Advanced Approaches for Wireless Sensor Network Applications and Cloud Analytics"

APPENDIX A

1.1 APPENDIX A.1: RLNC Simulation Algorithm

```
clear all
close all
clc
if(matlabpool('size'))
 matlabpool close force local;
end
matlabpool('open',6);

sim_start=datestr(now)
tic;

% Definitions
len_data=1000;     % Number of messages in data
p=2;               % Galois Field prime number p
GF_maxpow_m=6;     % Galois Field integer power
n_source=5;        % Number of source nodes
n_relay=6;         % Number of relay nodes
num_msg=1;
is_fullrank=0;

p_Err=0:0.01:1;
[dummy len_p_Err]=size(p_Err);
Sym_Err=0;

enc_vect_ones_bare = eye(n_relay,n_source);

for GF_m=1:GF_maxpow_m

parfor i_Err=1:len_p_Err            %Error Probability
Sym_Err=0;
Sym_Err_no_coding=0;
A_rec=[];
A_rec_no_coding=[];
n_error=0;

for cycle=1:(len_data/n_source)      %Message cycles
 enc_vects_bare=FR_gf_randi(0, 2^GF_m-1,
                            n_relay,n_source,GF_m);
 enc_vects=enc_vects_bare;
 enc_vect_ones=enc_vect_ones_bare;
```

```
 for i_era=1:n_relay
  for j_era=1:n_source
   if(rand(1)<p_Err(i_Err))
   enc_vects(i_era,j_era) = 0;
   enc_vect_ones(i_era,j_era) = 0;
  end
 end
end

if(rank(enc_vects)<n_source)
Sym_Err=Sym_Err+1;
end

if(rank(enc_vect_ones)<n_source)
Sym_Err_no_coding=Sym_Err_no_coding+1;
end
end

%Error probability
Sym_Err_Prob(i_Err,GF_m)=Sym_Err/(len_data/n_source);
Sym_Err_Prob_no_coding(i_Err,GF_m)=
                    Sym_Err_no_coding/(len_data/n_source);
end
end

movegui('northwest');
plot(p_Err,Sym_Err_Prob,'LineWidth',2);
title('Random␣Network␣Coding␣Scenario');
xlabel('Channel␣Erasure␣Probability');
ylabel('Symbol␣Error␣Probability');
grid on;
legend('m=1', 'm=2', 'm=3', 'm=4', 'm=5', 'm=6');
%hold on;
figure;
movegui('northeast');
plot(p_Err,Sym_Err_Prob_no_coding(:,1),'r','LineWidth',2);
title('Direct␣Transfer␣of␣Messages␣(No␣Network␣Coding)');
xlabel('Channel␣Erasure␣Probability');
ylabel('Symbol␣Error␣Probability(No␣Coding)');
grid on;

matlabpool close;

elapsed_sec=toc;
seconds2human(elapsed_sec)
sim_end=datestr(now)
```

1.2 APPENDIX A.2: SRDY Interrupt Event Registration Algorithm

```
wiced_gpio_input_irq_enable(
       ZNP_SRDY, IRQ_TRIGGER_BOTH_EDGES,
       SRDY_irq_handler, (void*) i );

wiced_rtos_create_worker_thread(
       &Worker1Thread,
       WICED_NETWORK_WORKER_PRIORITY,
       WORKER1_THREAD_STACK_SIZE, 10 );

void SRDY_irq_handler( void* arg )
       {
       wiced_rtos_send_asynchronous_event(
       &Worker1Thread,
       SRDY_Handler_Event, arg);
       }
```

1.3 APPENDIX A.3: SRDY Interrupt Handler Algorithm

```
wiced_result_t SRDY_Handler_Event( void* arg )
{
 switch ( IRQ_State )
 {
   case IDLE:
   {
     wiced_gpio_output_low( ZNP_MRDY );
     wiced_spi_transfer( &spi_ZNP, &POLL_seg[0], 1 );
     IRQ_State = POLLING;
     break;
   }
   case POLLING:
   {
     wiced_spi_transfer( &spi_ZNP, &POLL_seg[1], 1 );
     POLL_seg[2].length = POLL_frame[0];
     wiced_spi_transfer( &spi_ZNP, &POLL_seg[2], 1 );
     EnQueue( POLL_frame, IN );
     wiced_gpio_output_high( ZNP_MRDY );
     IRQ_State = IDLE;
     break;
   }
   case Start_SREQ:
   {
     SREQ_seg[0].tx_buffer = OutFrame;
     SREQ_seg[0].length = OutFrame[0] + 3;
     wiced_spi_transfer( &spi_ZNP, &SREQ_seg[0], 1 );
     IRQ_State = Wait_SRSP;
     break;
   }
```

```
    case Start_AREQ:
    {
      AREQ_seg.tx_buffer = AREQ_frame;
      AREQ_seg.length = AREQ_frame[0] + 3;
      wiced_spi_transfer( &spi_ZNP, &AREQ_seg, 1 );
      IRQ_State = AREQ_END;
      break;
    }
    case AREQ_END:
    {
      wiced_gpio_output_high( ZNP_MRDY );
      IRQ_State = IDLE;
      break;
    }
    case Wait_SRSP:
    {
      SREQ_seg[1].rx_buffer = SRSP_frame;
      wiced_spi_transfer( &spi_ZNP, &SREQ_seg[1], 1 );
      SREQ_seg[2].length = SRSP_frame[0];
      SREQ_seg[2].rx_buffer = &SRSP_frame[3];
      wiced_spi_transfer( &spi_ZNP, &SREQ_seg[2], 1 );
      EnQueue( SRSP_frame, IN );
      wiced_gpio_output_high( ZNP_MRDY );
      IRQ_State = IDLE;
      break;
    }
    default:
    {
      IRQ_State = IDLE;
      break;
    }
  }
return WICED_SUCCESS;
}

switch ( CurrentState )
{
  case IDLE:
  {
    if ( IRQ_State == IDLE )
      {
        else if ( !DeQueue( OutFrame, OUT ) )
        {
          CurrentState = Start_SREQ;
          EnQueue( ZB_SEND_DATA_REQUEST, OUT );
        }
      }
    break;
  }
```

```
  case Start_SREQ:
  {
    IRQ_State = Start_SREQ;
    wiced_gpio_output_low( ZNP_MRDY );
    CurrentState = IDLE;
    break;
  }
  default:
    break;
}
```

1.4 APPENDIX A.4: Circular Queue Init, Enqueue and Dequeue Algorithms

```
void InitQueues( void )
// Initializes enqueue and dequeue pointers.
{
  EnQptrIN = QbufferIN;
  DeQptrIN = QbufferIN;
  EOBptrIN = QbufferIN;
  EOBptrIN += QbufferSizeIN;
  NBavailIN = QbufferSizeIN;
  EnQptrOUT = QbufferOUT;
  DeQptrOUT = QbufferOUT;
  EOBptrOUT = QbufferOUT;
  EOBptrOUT += QbufferSizeOUT;
  NBavailOUT = QbufferSizeOUT;
}


unsigned char EnQueue(void *pDataIn, direction INorOUT)
{
  unsigned char NByte; // number of bytes
  unsigned char i; // byte index
  unsigned char *Bptr; // byte pointer
  NByte = *(unsigned char *) pDataIn;
  NByte += 3; // ZNP frame LENGTH + COMMAND part

  switch ( INorOUT )
  {
    case IN:
    {
      if ( NByte > NBavailIN ) // buffer overrun
        return (unsigned char) 1; // overrun error
      else
      {
        Bptr = (unsigned char *) pDataIn;
        for ( i = NByte; i > 0; i— )
        {
```

```
            *EnQptrIN = *Bptr;
            EnQptrIN++;
            if ( EnQptrIN == EOBptrIN )
              EnQptrIN = QbufferIN;
            Bptr++; // increment input pointer
          }
          NBavailIN -= NByte;
        }
        break;
      }

      case OUT:
      {
        if ( NByte > NBavailOUT ) buffer overrun
          return (unsigned char) 1; //overrun error
        else
        {
          Bptr = (unsigned char *) pDataIn;
          for ( i = NByte; i > 0; i-- )
          {
            *EnQptrOUT = *Bptr;
            EnQptrOUT++; // increment enqueue pointer
            if ( EnQptrOUT == EOBptrOUT )
              EnQptrOUT = QbufferOUT;
            Bptr++; // increment input pointer
          }
          NBavailOUT -= NByte;
        }
        break;
      }
    }
    return (unsigned char) 0;
} // end of function EnQueue

unsigned char DeQueue(void *pDataOut, direction INorOUT)
{
    unsigned char NByte; // number of bytes in data item
    unsigned char i; // byte index
    unsigned char *Bptr; // byte pointer

    switch ( INorOUT )
    {
      case IN:
      {
        if ( DeQptrIN == EnQptrIN )
          return (unsigned char) 1;
        else
        {
          NByte = *DeQptrIN;
```

```
          NByte += 3; // ZNP frame LENGTH + COMMAND part
          Bptr = (unsigned char *) pDataOut;
          for ( i = NByte; i > 0; i-- )
          {
            *Bptr = *DeQptrIN;
            DeQptrIN++; // increment dequeue pointer
            if ( DeQptrIN == EOBptrIN )
            DeQptrIN = QbufferIN;
            Bptr++; // increment output pointer
          }
          NBavailIN += NByte;
        }
        break;
      }

      case OUT:
      {
          if ( DeQptrOUT == EnQptrOUT )
            return (unsigned char) 1;
          else
          {
            NByte = *DeQptrOUT; // get the number of bytes
            NByte += 3; // ZNP frame LENGTH + COMMAND part
            Bptr = (unsigned char *) pDataOut;
            for ( i = NByte; i > 0; i-- )
            {
              *Bptr = *DeQptrOUT;
              DeQptrOUT++; // increment dequeue pointer
              if ( DeQptrOUT == EOBptrOUT )
                DeQptrOUT = QbufferOUT;
              Bptr++; // increment output pointer
            }
            NBavailOUT += NByte;
          }
          break;
       }
    }
    return (unsigned char) 0;
} // end of function DeQueue
```

1.5 APPENDIX A.5: XOR Coding Application Algorithm

```
static wiced_result_t send_msg( void *arg )
{
 for ( j = 0; j < NData; j++ )
  {
   Coded_Data[j] = DatafromA[j] ^ DatafromB[j] ;
  }
```

```
 OutData[0] = 8 + NData + CmdLen;
 OutData[10] = NData + CmdLen;

 for ( j = 1; j <= 9; j++ )
 {
  OutData[j] = ZB_SEND_DATA_REQUEST[j];
 }

 for ( j = 0; j < NData; j++ )
 {
  OutData[15 + j] = Coded_Data[j];
 }
 EnQueue( OutData, OUT );

 return WICED_SUCCESS;
 }
}
```

1.6 APPENDIX A.6: Publication on the thesis

Advanced Approaches for Wireless Sensor Network Applications and Cloud Analytics

Gorkem Ozvural
R&D Department
Vestel Electronics Inc.
MOSB Vestel City
45030 - Manisa, TURKEY
Email: gorkem.ozvural@vestel.com.tr

Gunes Karabulut Kurt
Department of Electronics and
Communication Engineering
Istanbul Technical University
34469 - Maslak, Istanbul, TURKEY
Email: gkurt@itu.edu.tr

***Abstract*—Although wireless sensor network applications are still at early stages of development in the industry, it is obvious that it will pervasively come true and billions of embedded microcomputers will become online for the purpose of remote sensing, actuation and sharing information. According to the estimations, there will be 50 billion connected *sensors* or *things* by the year 2020. As we are developing first to market wireless sensor-actuator network devices, we have chance to identify design parameters, define technical infrastructure and make an effort to meet scalable system requirements. In this manner, required research and development activities must involve several research directions such as massive scaling, creating information and big data, robustness, security, privacy and human-in-the-loop. In this study, wireless sensor networks and Internet of things concepts are not only investigated theoretically but also the proposed system is designed and implemented end-to-end. Low rate wireless personal area network sensor nodes with random network coding capability are used for remote sensing and actuation. Low throughput embedded IP gateway node is developed utilizing both random network coding at low rate wireless personal area network side and low overhead websocket protocol for cloud communications side. Service-oriented design pattern is proposed for wireless sensor network cloud data analytics.**

I. Introduction

It gradually becomes clear that wireless sensor network applications and Internet of things concept will reshape our environment, facilities, cities and our world in the upcoming decades [1]. With the help of new developments in wireless sensor networks (WSN) and Internet of things (IoT), novel application fields and scenarios will be formed according to the needs of human acitivities. There will be an explosion in the number of cloud services which are related to the usage of big data collected from WSN nodes. At this stage, several threats emerge for the scalability, manageability and efficiency of the overall system as the number of devices increase [2]. To ensure effective manageability and scalability, efficient design attitude should start from the most simplest unit to the most complex one in WSN system infrastructure. The efficiency in radio frequency (RF) transmission is not only about physical layer (PHY) but also all of the protocol layers over PHY. Network coding is a promising technique for using radio spectrum efficiently. If we can converge both network coding techniques in low rate wireless personal area networks (LR-WPAN) and service-oriented design in cloud analytics, we can achieve more efficient WSN ecosystem.

II. Background

A. Service-Oriented Architecture

Service-oriented design pattern originally evolved from the object-oriented analysis and design concepts [3][4]. In object-oriented analysis, the system functionality is isolated from the implementation constraints. Certain characteristics of each object can be abstracted according to the requirements and overall system can be managed more simply. Once the object paradigm is defined, inheritance-based design performs well to some extent, while component-based design pattern helps us to define objects in a more complex compositions. Service-oriented architecture comes up right here by defining a component which provides a black-box encapsulation of related services. This provides distribution of system funtionality and simplification of complex problems within each system.

B. Network Coding

The idea of network coding was first introduced at the beginning of 2000s, now there are lots of studies and implementations [5][6][7]. Let's begin with the famous example to summarize the idea behind network coding. In the scenario shown in Figure 1, Alice and Bob wants to exchange packets via a relay node in between. First, Alice sends her packet to relay node and then relay node forwards it to Bob. Thirdly, Bob sends his packet to relay node and fourth relay node forwards it to Alice. This data exchange method requires 4 cycles. Let's examine the scenario with network coding approach. First Alice sends her packet to relay node and second Bob sends his packet to relay node. In the third step, relay node exclusive ORs (XOR) the two packets and transmits the XORed version. Since Alice and Bob know their own packets, they simply XOR the received packet with their own packets and they obtain each other's packet. The packet exchange ends with 3 cycles and saved cycles can be used for new packet transmissions which increases throughput.

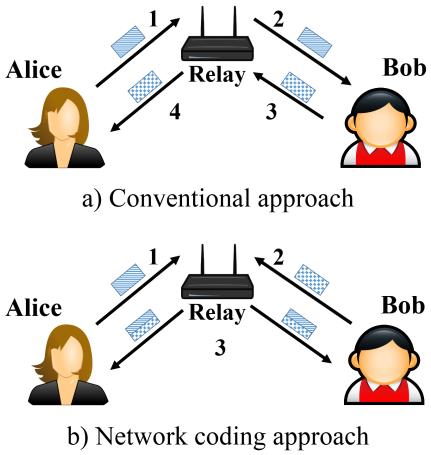


Fig. 1. Conventional (a) vs. network coding (b) approach. 3 transmissions instead of 4 saves 1 transmission cycle for any other data exchange.

Network coding is a promising method for future wireless networks and it has lots of advantages. Some of the most important IoT related advantages of network coding are low complexity for near-optimal routing [7][8][9][10], throughput enhancement [5], robustness to link failures and packet losses [8] [11].

C. WebSocket Protocol

Historically, web applications which needs bidirectional communication between client and server has used inefficient polling techniques for updates from server while sending upstream messages as a distinct hyper-text transfer protocol (HTTP) calls. This caused forcing the server to open more than one transmission control protocol (TCP) sockets for each client, high overhead messages for each client-to-server HTTP calls and keeping the mapping from each outgoing connections to the incoming connection to track replies. A simpler solution could be to use single TCP connection for full-duplex communication between client and server. This is the underlying idea of WebSocket protocol. The WebSocket protocol is designed to overcome technical insufficiencies of existing HTTP based bidirectional communication technologies while exploiting existing infrastructure such as proxies, filtering and authentication. It works over HTTP ports 80 and 443 to support HTTP proxies and intermediaries. The protocol has two main states called handshake and the data transfer. As an example, the handshake message from client to server looks as follows:

GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13

Once the handshake between client and server successfully ended, a two-way communication channel is established over one TCP connection and the data transfer part starts. Now each side can send data frames at will without waiting the other party. As the last detail for a brief introduction to WebSocket protocol, there are some specific frames *ping* and *pong*, used between client and server to keep TCP connection alive by preventing intermediary network devices closing the connection because of timeouts.

III. System Realization

A. LR-WPAN Embedded IP Gateway

WSN applications generally exploit from low throughput LR-WPAN protocols. One of the most widespread protocols that uses IEEE 802.15.4 LR-WPAN is ZigBee. Mesh network capability, low-power consumption and open source availability makes ZigBee preferable to other technologies. However, ZigBee protocol, as other LR-WPAN protocols, needs a gateway device to be able to communicate with the devices in IP domain. Thanks to the low cost high performance embedded microcontrollers, implementing embedded IP gateway for a ZigBee network becomes an attainable solution. As one alternative of quiet a lot solutions, we preferred Broadcom's brand-new wireless Internet connectivity for embedded devices (WICED) platform in combination with TI's CC2530 ZigBee network processor (ZNP) to realize LR-WPAN embedded IP gateway. In this combination, CC2530 has a low power 8051 core while WICED module has ARM Cortex-M3 microprocessor. Since, 8051 core in CC2530 with limited processing power, RAM and ROM size won't be suitable for being the main processor of gateway node, ARM Cortex-M3 core in WICED system in package (SiP) is chosen as the central processor.

WICED platform comes with two alternatives for IP stack and real time operating system (RTOS): NetX or LwIP TCP/IP stacks and ThreadX or FreeRTOS embedded operating systems. In our application, ARM Cortex-M3 reduced instruction set computing (RISC) microprocessor using ThreadX or FreeRTOS @120MHz on WICED is more suitable for handling IP network interfacing and forwarding tasks of the gateway node, while ZNP (ZigBee Network Processor) is in charge of interfacing ZigBee network. Proposed system architecture can be seen in Figure 2. At the right hand side, ZNP system architecure is shown. It uses IEEE 802.15.4 PHY and medium access control (MAC) layers under ZigBee network and application layers. ZNP has a serial peripheral interface (SPI) to communicate with the host processor. At the left hand side, WICED system is shown. WICED features IEEE 802.11n PHY and MAC layers under transport and network layers. Over the transport layer, WebSocket protocol operates as one of the low overhead, full-duplex communication protocols over a single TCP connection.

When we think about two wireless technologies at the same node, we have to consider coexistence analysis. IEEE 802.15.4 Sub-1 GHz ZigBee modules combined with a 2.4 GHz or 5

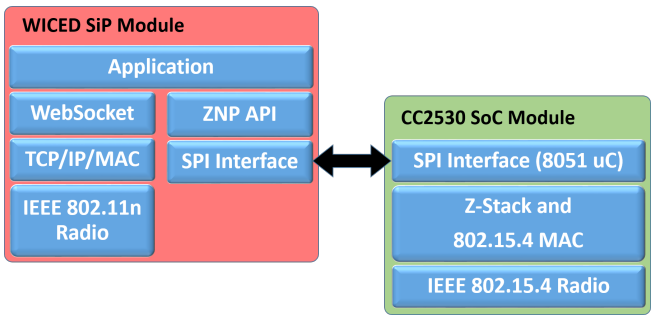


Fig. 2. LR-WPAN embedded IP gateway system architecture: WICED host MCU and CC2530-ZNP combination.

GHz WiFi module doesn't have any coexistence issues. The problem occurs when both of them use the same 2.4 GHz band. If we think about using 2.4 GHz band for both ZigBee and WiFi, we have to consider co-channel interference which will increase PER and retransmissions. In [13][14][15], researchers analyzed coexistence of WiFi and ZigBee technologies by applying several coexistence scenarios in a home environment.

B. WSAN Devices and Network Coding

To form a wireless sensor-actuator network (WSAN) environment, multipurpose WSAN nodes are developed in the study. These nodes are equipped with relays and sensors, making it possible to connect them on any kind of electrical device for turning it on and off, collecting data from its environment etc. ZigBee network processor in combination with ARM Cortex-M3 microcontoller is used for LR-WPAN communications part. Network coding is proposed and implemented between ZigBee nodes including embedded IP gateway. Because of the variable topology property of WSNs, random network coding is the most appropriate technique for increasing reliability, throughput and battery life for WSN devices. Figure 3 denotes ZigBee WSAN.

The core idea of network coding is to allow scrambling data at intermediate nodes [5] by generating a linear combination of the incoming packets.

Consider the butterfly network shown in Figure 4. Simply addition of the bits of incoming packets in $GaloisField(2)$ for coding purposes means bitwise XOR. For practical consideration, the Relay node in the middle uses XOR operation for encoding the packets while Sink-1 and Sink-2 nodes use XOR operation for decoding the packets.
Let's take the network coding operation more officially. Consider original packet A as:

$$\mathbf{A} = \{A_1, A_2, ..., A_k\} \tag{1}$$

Choosing coding coefficients randomly, the encoding vector can be represented as:

$$\mathbf{e} = \{e_1, e_2, ..., e_k\} \tag{2}$$

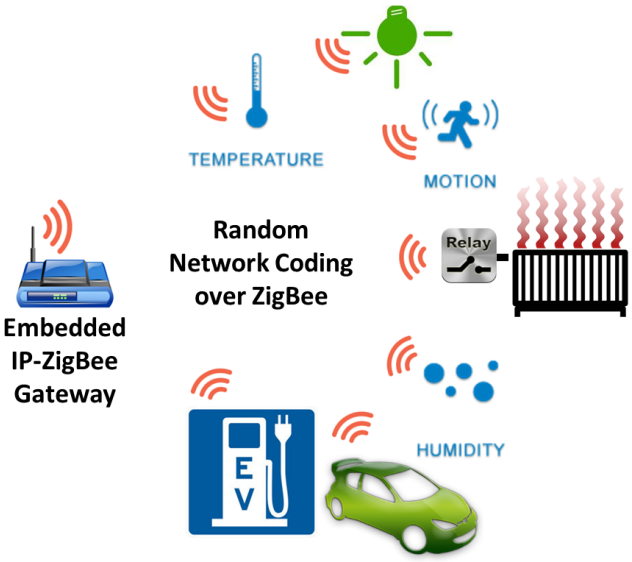


Fig. 3. Random network coding over ZigBee WSAN and smarthome sensors and actuators.

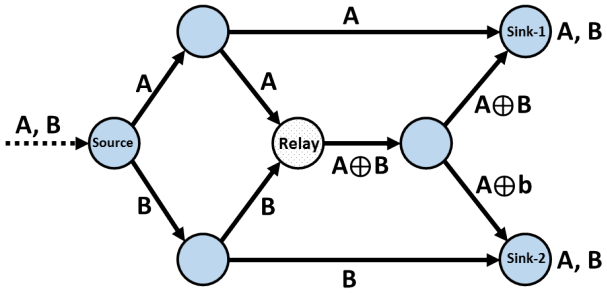


Fig. 4. The butterfly network. Each directed link in this network is capable of transferring a single packet in one cycle. There are 2 data packets, A and B, which must be transferred to the Sink-1 and Sink-2 nodes.

We can generate the linear combination of incoming packet A in relay node and transmit it to the sink node:

$$\mathbf{X} = e_1A_1 + e_2A_2 + ... + e_kA_k \tag{3}$$

Considering the packet recovery problem from receiver side, let's denote the received packet R_i in the i^{th} transmission cycle as:

$$\mathbf{R_i} = \{a_{i1}A_1 + a_{i2}A_2 + ... + a_{ik}A_k\} \tag{4}$$

Receiver gets the coded packets successively and keeps a received packet matrix:

$$\mathbf{R} = \begin{bmatrix} a_{11}A_1 & a_{12}A_2 & . & . & a_{1k}A_k \\ a_{21}A_1 & a_{22}A_2 & . & . & a_{2k}A_k \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{k1}A_1 & a_{k2}A_2 & . & . & a_{kk}A_k \end{bmatrix}$$

Applying Gaussian elimination method to the received packet matrix, we can achieve a solvable linear system. Then it becomes easy to obtain recovered message array A_r:

$$\mathbf{Ar} = \{Ar_1, Ar_2, ..., Ar_k\} \tag{5}$$

C. Service-Oriented Cloud Architecture

To enable these end devices to send and receive data over the Internet, we need to define overall system architecture including cloud side. By definition, IoT paves the way for pervasive scaling of Internet enabled devices. Therefore, it should have expandable structure in the aspect of both simultaneously served devices and number of associated services to those devices. Furthermore, service oriented architecture [3][4] design pattern should be applied for the efficient scaling of the value added services (VAS) cloud analytics. The architecture that meets all of those needs is defined in Figure 5.

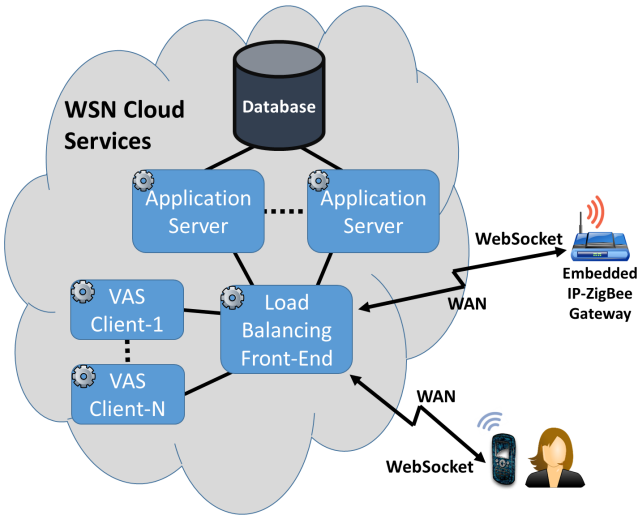


Fig. 5. Expandable overall system architecture including value-added cloud services.

On the right hand side of Figure 5, IoT device side implementation is represented by a smarthome sensor network. There is a gateway node in between LR-WPAN sensors and IP network infrastructure. Regardless of LR-WPAN and wide area network (WAN) protocols, there must be a kind of packet routing and forwarding mechanism in between. On the left hand side, proposed system for IoT cloud side is shown. This architecture is implemented for testing the operation of LR-WPAN network coding mechanism.

D. Cloud Data Analytics

Data analytics is not a new concept but it changed its shell with emerging WSN and IoT applications. With the increase in the number of sensors, more and more data is now collected each and every second. Smart sensing and actuation started to be used for agriculture and stockbreeding, home security, smart diagnosis, vending machine and production automation applications. The need of distinguishing useful data from this big data collection emerged and data mining techniques started to be implemented for the cloud analytics. The service oriented cloud architecture proposed in this study takes those cloud applications as value added services and optimizes the cloud architecture according to the requirements of massive scaling nature of WSNs.

IV. Conclusion

In this study, new and promising technique, random network coding, is proposed to the WSN applications using LR-WPAN ZigBee protocol. Embedded IP gateway is realized with a combined node using ZigBee and WiFi embedded modules. Low overhead, low latency and low resource WebSocket protocol is used between embedded gateway and cloud applications server. Service-oriented design is proposed and implemented to the cloud services, enabling incremental system scaling for increasing number of sensor devices and related value added services.

WSN and IoT ecosystem will form the future of networked society. The proposed system architecture is highly applicable for smart-home, smart-office, smart-grid and smart-city applications with integrated value added services. Utilization of decreased transmission cycles as an outcome of network coding techniques and efficient service-oriented system architecture provides both manageability and multiplatform compatibility to future WSN applications.

Acknowledgment

This work is supported by TUBITAK under Grant 113E294 and Vestel Electronics Inc.

References

[1] Dave Evans, *The Internet of Things How the Next Evolution of the Internet Is Changing Everything*, White Paper, Cisco Internet Business Solutions Group (IBSG), 2011.

[2] John A. Stankovic, *Research Directions for the Internet of Things*, Vol:1, No:1, IEEE Internet of Things Journal, 2014.

[3] Mark Endrei and Jenny Ang and Ali Arsanjani and Sook Chua and Philippe Comte and Pl Krogdahl and Min Luo and Tony Newling, *Patterns: Service-Oriented Architecture and Web Services*, Redbooks, IBM, 2004.

[4] Goh Chun Lin and Koh Eng Tat Desmond and Naing Tayza Htoon and Nguyen Van Thuat, *A Fresh Graduates Guide to Software Development Tools and Technologies, Chapter-10: Service Oriented Architecture*, School of Computing, National University of Singapore, 2011.

[5] Rudolf Ahlswede and Ning Cai and Shuo-Yen Robert Li and Raymond W. Yeung, *Network information flow*, Vol:46, No:4, IEEE Transactions on Information Theory, 2000.

[6] S. Deb and M. Effros and T. Ho and D. R. Karger and R. Koetter and D. S. Lun and M. Medard and N. Ratnakar, *Network coding for wireless applications: A brief tutorial*, IWWAN, London, UK: 2005.

[7] T. Ho and R. Koetter and M. Medard and D. Karger and M. Effros, *The Benefits of Coding over Routing in a Randomized Setting*, ISIT, Yokohama, Japan, 2003.

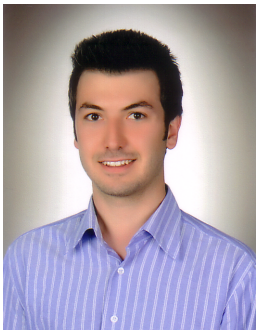
[8] Tracey Ho and Desmund S. Lun, *Network Coding*, Cambridge, England: Cambridge University Press, 2008.

[9] S. Deb and M. Medard and C. Choute, *Algebraic gossip: A network coding approach to optimal multiple rumor mongering*, Vol:52, No:6, IEEE Transactions on Information Theory, 2006.

[10] S. Katti and D. Katabi and W. Hu and H. Rahul and M. Medard, *The importance of being opportunistic: Practical network coding for wireless environments*, 43rd Annual Allerton Conference on Communication, Control and Computing, 2005.

[11] R. Koetter and Muriel Medard, *An algebraic approach to network coding*, Vol:11, No:5, IEEE/ACM Transactions on Networking, 2003.
[12] I. Fette and A. Melnikov, *The WebSocket Protocol, Request for Comments: 6455*, Internet Engineering Task Force (IETF), ISSN:2070-1721, 2011.
[13] F. Dominguez and A. Touhafi and J. Tiete and K. Steenhaut, *Coexistence with WiFi for a Home Automation ZigBee product*, IEEE 19th Symposium on Communications and Vehicular Technology, 2012.
[14] T. Han and B. Han and L. Zhang and X. Zhang and D. Yang, *Coexistence study for WiFi and ZigBee under smart home scenarios*, IEEE 3rd International Conference on Network Infrastructure and Digital Content, 2012.
[15] Xinyu Zhang and Kang G. Shin, *Cooperative Carrier Signaling: Harmonizing Coexisting WPAN and WLAN Devices*, Vol:21, No:2, IEEE/ACM Transactions on Networking, 2013.

CURRICULUM VITAE



Name Surname: Görkem ÖZVURAL
Place and Date of Birth: Izmir - 09.01.1988

Address: Karacaoğlan Mahallesi 6240 Sokak No:16 K:5 D:19 Bornova İzmir - TURKEY

E-Mail: gorkemozvural@gmail.com

Education
MBA: Ozyegin University (2013-)
MSc: Istanbul Technical University (2011-)
BSc: Izmir Institute of Technology (2006-2011)

Training
10/2011-05/2012 ITU Cisco Networking Academy (İstanbul)
07/2007-11/2007 Deutsche Akademie für Sprachen (İzmir)
07/2004-08/2004 EF Cambridge Clare College (Cambridge)

Professional Experience:
10/2012- Senior System Design Engineer - Vestel Electronics Inc.
03/2011-08/2011 Crea Academy Intern - Ericsson
08/2010-10/2010 Radio Network Planning Intern - Turkcell
06/2010-07/2010 Military Communications Project Intern - Aselsan
06/2009-07/2009 Radio Network Optimization Intern - Iris Telecom

Languages:
English Listening:(C1) Reading:(C2) Spoken Interaction:(C1) Writing:(C1)
German Listening:(A2) Reading:(A2) Spoken Interaction:(A1) Writing:(A1)

List of Publications and Patents:
- **Özvural G.**, Gümüştekin Ş., Active Eye-Glasses with Local Dimming, *7th International Conference on Electrical and Electronics Engineering*, Bursa, Turkey, December 1-4, 2011.

PUBLICATIONS/PRESENTATIONS ON THE THESIS
- **Özvural G.**, Karabulut Kurt G., Advanced Approaches for Wireless Sensor Network Applications and Cloud Analytics, *7th Workshop on Emerging Wireless Sensors Network Applications*, Singapore, April 7-9, 2015.

COMPETITIONS PARTICIPATED AND AWARDS
11-12 April 2015 Austrian Society (INNOC) RobotChallenge 2015 3rd Prize
26-27 October 2014 IZTECH RoboLeague Innovative Category 1st Prize
2012 Informatic Association of Turkey & Turkcell Academy, Technology Leaders

Master's Scholarship 2012
2011 Ericsson Crea Academy Scholarship
14 May 2005 Özel Ege Lisesi Project Competition Physics Category 3rd Prize
12-13 May 2007 Boğaziçi University Robotic Days Sumo Robot Category 3rd Prize
24-25 May 2008 Ege Bölgesi Sanayi Odası Project and Robot Competition Special Jury Prize