İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

IN IEEE 802.15.4 STANDARD GUARANTEED TIME SLOT PERFORMANCE, SYNCHRONOUS DATA ACQUISITION AND SYNCHRONIZATION ERROR

M.Sc. Thesis by
Cengiz GEZER

Department : Electronics and Communication Engineering
Programme: Telecommunication Engineering

JUNE 2008

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

IN IEEE 802.15.4 STANDARD GUARANTEED TIME SLOT PERFORMANCE, SYNCHRONOUS DATA ACQUISITION AND SYNCHRONIZATION ERROR

M.Sc. Thesis by
Cengiz GEZER
504051304

Date of submission : 2 May 2008
Date of defence examination: 9 June 2008

Supervisor (Chairman): Prof. Dr. Tayfun AKGÜL

Members of the Examining Committee Assoc. Prof.Dr. Selçuk PAKER

Assis. Prof.Dr. Feza BUZLUCA

JUNE 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

IEEE 802.15.4 STANDARDINDA GARANTİLENMIŞ ZAMAN DİLİMİ BAŞARIMI, ALGILAYICI DÜĞÜMLERİ İLE EŞZAMANLI VERİ EDİNME VE EŞZAMANLAMA HATASI

YÜKSEK LİSANS TEZİ

Cengiz GEZER
504051304

Tezin Enstitüye Verildiği Tarih : 2 Mayıs 2008
Tezin Savunulduğu Tarih : 9 Haziran 2008

Tez Danışmanı :	**Prof.Dr. Tayfun AKGÜL**
Diğer Jüri Üyeleri	**Doç. Dr. Selçuk PAKER**
	Yard. Doç Dr. Feza BUZLUCA

HAZİRAN 2008

PREFACE

I would like to thank Prof. Roberto VERDONE and my colleague Federico SCUDELLARI for their support, guidance and ideas during my studies in WiLab Laboratory of University of Bologna. They provided me a peaceful and enjoyable studying environment. I would also like to express my gratitude to my supervisor, Prof. Tayfun AKGÜL, especially for his positive and helpful attitude towards me. Alexander DEIERLING and Dimitra DEVELEGKA have always been next to me to ready to help during my studies. Thank you for your great friendship at the student residence. I should also thank to Ufuk ÜLÜĞ because of his valuable support on the bureaucratic procedures. When I was feeling need to tell my achievements to someone Süleyman BAYKUT was the only person who can both understand and appreciate them. Thank you for your sincerity and help. Last but not least I would like to thank my mother. I have opportunities because of you.

CONTENTS

LIST OF ABBREVIATIONS

ADC : Analog Digital Converter
API : Application Programming Interface
ASP : Application Support Package
BI : Beacon Interval
BO : Beacon Order
BPSK : Binary Phase-Shift Keying
BSN : Beacon Sequence Number
CAP : Contention Access Period
CCF : Conversion Complete Flag
CDF : Cumulative Distribution Function
CFP : Contention-Free Period
CMOS : Complementary Metal Oxide Semiconductor
CRC : Cyclic Redundancy Check
CSMA-CA : Carrier Sense Multiple Access with Collision Avoidance
DSSS : Direct Sequence Spread Spectrum
ECMA : European Computer Manufacturers Association
ETSI : European Telecommunications Standards Institute
FCS : Frame Check Sequence
FDD : Frequency Division Duplex
FFD : Full-Function Device
FHSS : Frequency Hopping Spread Spectrum
FIFO : First In, First Out
FPGA : Field Programmable Gate Array
GPS : Global Positioning System
GTS : Guaranteed Time Slot
GUI : Graphical User Interface
HR-WPAN : High Data Rate Wireless PAN
IFS : Interframe Space or Spacing
IEEE : Institute of Electrical and Electronics Engineers
ISM : Industrial, Scientific, and Medical
LAN : Local Area Network
LIFS : Long Interframe Spacing
LDPC : Low-Density Parity-Check
LNA : Low Noise Amplifier
LQ : Link Quality
LQI : Link Quality Indication
LR-WPAN : Low-Rate Wireless Personal Area Network
MAC : Medium Access Control
MAN : Metropolitan Area Network
MCPS : MAC Common Part Sublayer
MCPS-SAP : MAC Common Part Sublayer-Service Access Point

MCU : Microcontroller Unit
MEMS : Micro-Electro-Mechanical System
MFR : MAC Footer
MHR : MAC Header
MLME : MAC Sublayer Management Entity
MLME-SAP : MAC Sublayer Management Entity-Service Access Point
MPDU : MAC Protocol Data Unit
MSDU : MAC Service Data Unit
NLOS : Non-Line of Sight Operation
NTP : Network Time Protocol
NWK : Network Layer
O-QPSK : Offset Quadrature Phase-Shift Keying
OFDM : Orthogonal Frequency Division Multiplexing
QAM : Quadrature Amplitude Modulation
QoS : Quality of Service
PAN : Personal Area Network
PC : Personal Computer
PDA : Personal Digital Assistant
PDF : Probability density Function
PDU : Protocol Data Unit
PHY : Physical Layer
PIB : PAN Information Base
POS : Personal Operating Space
PPDU : PHY Protocol Data Unit
PSDU : PHY Service Data Unit
RAM : Random Access Memory
RBS : Reference-Broadcast Synchronization
RF : Radio Frequency
RFD : Reduced-Function Device
SAP : Service Access Point
SD : Superframe Duration
SHR : Synchronization Header
SIFS : Short Interframe Spacing
SNR : Signal-to-Noise Ratio
SO : Superframe Order
TC : Turbo Code
TDD : Time Division Duplex
TOF : Timer Overflow Flag
TOIE : Timer Overflow Interrupt Enable
UWB : Ultra Wide Band
WAN : Wide area network
WPAN : Wireless Personal Area Network
WSN : Wireless Sensor Network

LIST OF FIGURES

LIST OF TABLES

IEEE 802.15.4 STANDARDINDA GARANTİLENMIŞ ZAMAN DİLİMİ BAŞARIMI, ALGILAYICI DÜĞÜMLERİ İLE EŞZAMANLI VERİ EDİNME VE EŞZAMANLAMA HATASI

ÖZET

Düşük fiyatları ile kolayca temin edilebilir olan Kablosuz Algılayıcı Ağları (KAA), yaygın hale gelmeye başlamıştır. Gelecek yıllarda, temin ve fiyat bakımından daha kolay elde edilebilir olacaklarını söylemek pek de yanlış bir tahmin sayılmaz. Bazı algılayıcı ağı uygulamalarında belirli bir aygıta düşük gecikme ve ayrılmış bant genişliği tanımak belirgin bir öneme sahip olabilir. Bu tür gereksinimlere, IEEE 802.15.4 Standardında tanımlanmış Garantilenmiş Zaman Dilimi (GZD) mekanizması çözüm sağlar. Bu çalışmada, Freescale Yarıiletken tarafından üretilen 13192 EVK ile GZD başarımı ölçülmüştür. Ölçülen başarım, kuramsal üretilen iş (throughput) ve kuramsal en büyük yararlı iş (goodput) değerleri ile kıyaslanmıştır. Başarım ölçümlerinin yanında, iki algılayıcı düğümü kullanılarak eş zamanlı veri edinme de başarıyla gerçekleştirilmiştir. Edinilmiş veriler eşgüdümleyiciye aktarılırken GZD kullanılmıştır. Ayrıca başarım ölçümlerinden elde edilen sonuçlar yardımı ile iki algılayıcı düğümün eşzamanlı hale getirilme ayarlamaları yapılmıştır. Algılayıcı düğümlerinin eşzamanlaması veya KAA'nda gerçekleşen olayların zaman sırası, geliş doğrultusu (direction of arrival), dizi işaret işleme (array signal processing) veya gözetim (surveillance) uygulamalarında dikkatlice gözden geçirilmesi gereken önemli bir olgudur. Bir eşzamanlama yöntemine ulaşabilmek için geliştirilen uygulamada IEEE 802.15.4 Standardında tanımlı parıldak haber göstergesi ilkeli (beacon notify indication primitive) kullanılmıştır. Ayrıca düğümler arası eşzamanlama hatası incelenmiştir.

IN IEEE 802.15.4 STANDARD GUARANTEED TIME SLOT PERFORMANCE, SYNCHRONOUS DATA ACQUISITION WITH SENSOR NODES AND SYNCHRONIZATION ERROR

ABSTRACT

Wireless Sensor Networks (WSN) is getting wide-spread attention since they became easily accessible with their low costs. Predicting that they will be more accessible and cheaper than now in next few years will not be a faulty forecast. In some of the sensor network applications, low latency and reserved bandwidth to a particular device may have a significant importance. Guaranteed Time Slot (GTS) mechanism defined in IEEE 802.15.4 Standard provides solutions for such necessities. In this study, performance of GTS is measured on 13192 Evolution Kit modules from Freescale Semiconductor. This performance is compared with the theoretical throughput and maximum goodput values. Besides the performance measurements, synchronous data acquisition with two sensor nodes has been successfully realized. While transmitting acquisition data to the coordinator GTS is used. Furthermore, obtained results from the performance measurements used for tuning the synchronization of two nodes. Synchronization of sensor nodes or chronologically sorting the events happened in a WSN is an important phenomenon that need to be examined carefully for direction of arrival, array signal processing or surveillance applications. In order to find a synchronization scheme beacon notification indication primitive defined in the 802.15.4 standard has been used in the developed applications. Also synchronization error introduced by the nodes is inspected.

1 INTRODUCTION

Wireless Sensor Networks (WSN) is an emerging field that needs more research and investment. In the near future, we will be living with cheap sensors everywhere and these sensors will constitute the wireless sensor networks that surround us. In WSNs, IEEE 802.15.4 standard can be implemented. This study mainly concerns to reveal and examine the application performance of IEEE 802.15.4 for low latency communication and sensor arrays. It is an experimental work with an evolution kit from Freescale Semiconductor. Guaranteed Time Slots (GTS) is on the main focus of the study. First step of this study was to measure the performance of the GTS in a laboratory environment and after that with the result of the measurements synchronously acquired data from two different sensors are sent to a coordinator device by using GTS. Time synchronization of the sensor data is made by using the beacons defined in IEEE 802.15.4 standard. But still GTS is the main concern since sensor data needs low latency.

Second part of this study contains definition of WSN and application areas of it with some real-life applications. In the third part of the study to constitute a concrete understanding about the 802.15.4 standard other standards from IEEE are examined and the position of the 802.15.4 among them tried to be investigated. Interference in the ISM band is highlighted before measuring the performance of GTS. Finally, in the third part physical and MAC layers are explained. While MAC layer is being explained, GTS was emphasized. In the fourth part used hardware is briefly discussed. In the fifth part in which hardware setup and with what sort of software, goodput measurements are done is described. Moreover, developed codes for devices and the MATLAB graphical user interface are explained. In the sixth part by using the GTSs and beacon notification, developed application which acquires time synchronized data from two sensor cards are explained. Also in this part synchronization error introduced by the devices is inspected.

2 WIRELESS SENSOR NETWORKS:

A Wireless Sensor Network can be described as a sensor network consisting of densely distributed autonomous devices (nodes), using sensors to cooperatively monitor physical or environmental conditions and RF waves to send the monitored data to the base stations or coordinators [1].

The nodes in WSNs contain RF components, actuators, sensors and CMOS type electronic devices (interface and data fusion circuitry, specialized and general purpose signal processing units, and microcontrollers). These components are named together as Micro-Electro-Mechanical System (MEMS). In the latest development on the MEMS's allowed the production of low-cost, low-power, multifunctional sensor nodes. Nowadays the availability of cheap sensor nodes are enabling the application of distributed wireless sensing to be realized commercially.

When it is compared with the traditional sensors, WSNs yields improved line of sight and SNR because of its way of deployment and processing. Traditional sensors are deployed in the following two ways [2]:

- Large, complex sensor systems are usually deployed very far away from the phenomena to be sensed, and employ complex signal processing algorithms to separate targets from environmental noise.
- Carefully engineered network of sensors is deployed in the field, but individual sensors do not possess computation capability, instead transmitting time series of the sensed phenomena to one or more nodes which perform the data reduction and filtering.

On the other hand, WSNs are densely deployed either inside the phenomenon or very close to it and they are capable of carrying out simple computations. Three important concepts constitute the phenomenon of WSN are; distributed sensing, wireless communication and distributed processing [3].

- **Distributed Sensing:** The positions of sensor nodes need not be pre-determined through engineering calculations when the precise location of a signal of interest is unknown across the monitored region. Distributing them randomly yields higher SNR, and improved line of sight than having a single very sensitive sensor in one particular location. It is obvious that a distributed network of sensors will collect significantly different information than a system relying on a single sensor.
- **Wireless Communication:** In WSN applications the environment being monitored generally lacks of infrastructure for communications or energy, therefore untethered nodes must be supplied from local and finite energy sources, as well as rely on wireless communication channels to send data packets to each other.
- **Distributed Processing:** Finite energy budget of the untethered nodes restricts the design of WSNs. Communication is a key energy consumer as the radio signal power in sensor networks drops off with r^4 [3] due to ground reflections from short antenna heights. Therefore, it is desired to process data as much as possible inside the nodes to reduce the number of bits transmitted. Distributed processing shows itself as a solution to energy constraints in WSNs.

2.1 Sensor Networks Applications

Sensor networks can be built up from many different types of sensors which are able to monitor different kinds of ambient conditions such as [4]:

- Temperature
- Humidity
- Vehicular movement
- Lightning condition
- Pressure
- Soil makeup
- Noise levels
- The presence or absence of certain kinds of objects
- Mechanical stress levels on attached objects
- The current characteristics such as speed, direction, and size of an object

Application areas of WSNs can be categorized considering the application type [1]:

- Military applications
    - Monitoring friendly forces, equipment and ammunition
    - Battlefield surveillance
    - Reconnaissance of opposing forces and terrain
    - Targeting
    - Battle damage assessment
    - Nuclear, biological and chemical attack detection and reconnaissance
- Environmental applications
    - Forest fire detection
    - Biocomplexity mapping of the environment[5]
    - Flood detection[6]
    - Precision Agriculture
- Health applications
    - Telemonitoring of human physiological data [7]
    - Tracking and monitoring doctors and patients inside a hospital
    - Drug administration in hospitals [8]
- Home applications
    - Home automation [9]
    - Smart environment [10]
- Other commercial applications
    - Environmental control in office buildings
    - Interactive museums
    - Detecting and monitoring car thefts
    - Managing inventory control
    - Vehicle tracking and detection [11]

In order to constitute a concrete understanding of WSNs and the sensors used in, some of the real-life applications are given in the following pages.

2.1.1 Bird Observation on Great Duck Island

On Great Duck Island, Maine, United States, a WSN is being used to observe the breeding behaviour of a small bird called Leach's Storm Petrel [12]. The Sensor nodes for collecting data about humidity, pressure, temperature, and ambient light

level are installed inside the nesting burrows. System architecture can be seen in Figure 2.1. The sensor nodes transmit their data through the sensor network to the sensor network gateway since the nodes are deployed in dense patches that are widely separated. The gateway is responsible for transmitting sensor data from the sensor patch through a local transit network to the remote base station which has WAN connectivity and data logging capability. The base station connects to database across the internet. Finally, the data is displayed to the users through a user interface.

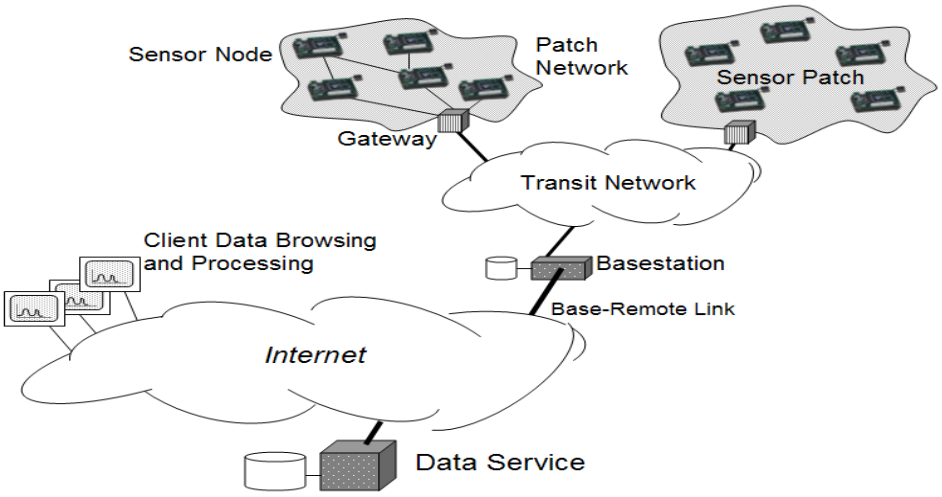

Figure 2.1: System Architecture for Leach's Storm Petrel Observation

2.1.2 Cattle Herding

At Cobb Hill Farms in Vermont, USA a WSN is being used to implement virtual fences. In this network cows are stimulated with an acoustic source when they try to cross a virtual fence line [13]. Each sensor in the WSN node contains a smart collar with a GPS unit, a Zaurus PDA, wireless networking, and a sound amplifier for providing acoustic stimuli to the cattle. Such a system can reduce the operating costs of installing and moving physical fences, and improve the usage of feedlots.

2.1.3 Ocean Water Monitoring

The ARGO project [14] is using a global array of 3,000 free-drifting profiling floating nodes to observe the temperature, salinity, and current profile of the upper 2000 m of the ocean. The goal is a quantitative description of the state of the upper ocean and the patterns of ocean climate variability, including heat and freshwater storage and transport. Locations of the ARGO Nodes around the World are in the Figure 2.2.

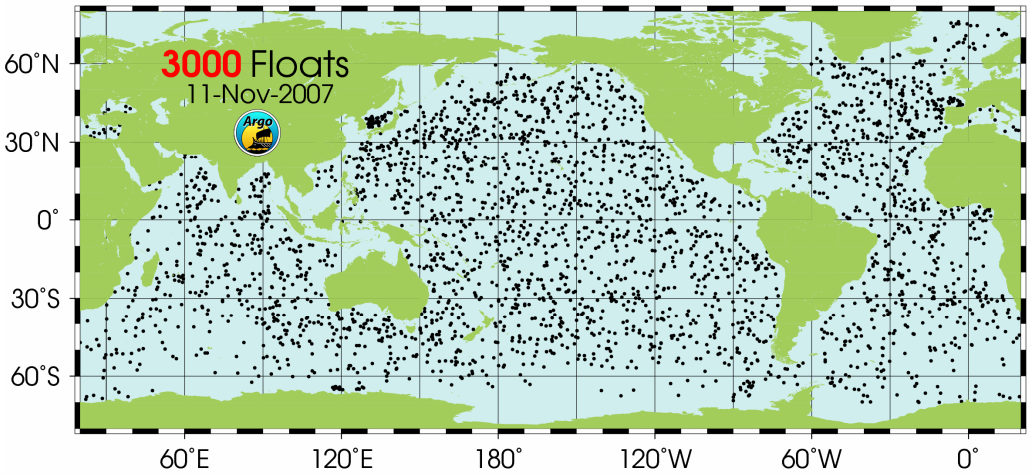

Figure 2.2: Locations of ARGO Nodes around the World

2.1.4 Grape Monitoring

In Oregon, United States a WSN is being used to monitor the conditions such as temperature, soil moisture, light, and humidity across a large vineyard [15]. Sensor nodes are deployed across a vineyard in a regular grid about 20 m apart. The sensor nodes form a two-tier multihop network, with nodes in the second tier sending data to a node in the first tier. Main goals of this WSN are harvesting the areas as soon as possible when the grapes in it are ripe, adapting the water/fertilizer/pesticide supply to the needs of individual grapes, protecting against frost, predicting insect/pest/fungi development, and developing new agricultural models.

2.1.5 Rescue of Avalanche Victims

A WSN is being used in saving people buried in avalanches [16]. For this purpose, skiers, snowboarders, hikers and the other people that may be in risk carry a sensor node that measures the oxygen level in blood and contains an accelerometer to derive the orientation of the victim. The aim is to better locate buried people and to limit overall damage by giving the rescue team indications of the state of the victims

2.1.6 Sniper Localization

A WSN is being used to locate snipers and the trajectory of bullets [17]. The system consists of acoustic sensor nodes that measure the muzzle blast and shockwave. The sensor nodes form a multihop ad hoc network. By comparing the time of arrival at distributed sensor nodes, the location of sniper can be found. The sensor nodes use a field programmable gate array (FPGA) chip to carry out the complex signal processing functions.

3 IEEE 802.15.4 STANDARD

IEEE 802.15.4 is a standard specially used for Low Rate Wireless Personal Area Networks (LR-WPAN). The main purpose of this standard is to provide ultra low complexity, ultra low power consumption and extremely low cost wireless networking solution in low data rate networks within the Personal Operating Space (POS). POS is a region with a radius of 10 meters.

Besides the 802.15.4 Standard, many wireless standards from IEEE are being developed continuously. In order to understand the reason why there are plenty of different standards for wireless networks, it is convenient to investigate the relationship among the frequency, range, bandwidth, and antenna size.

3.1 Relationship between Frequency, Range, Bandwidth, and Antenna Size

In a fixed energy level the frequency in wireless communication determines physical characteristics of the signal [18]. For example:

- Low frequencies can penetrate longer range than high frequencies.
- High frequencies can carry more information than low frequencies.
- Required size of antenna to tune the signal is large for low frequencies and small for high frequencies.
- Low frequency is more suitable for Non-Line of Sight Operation (NLOS) than high frequency.

Necessity of range, bandwidth, and antenna size of a particular application determines the frequency. These demands are contradictory and an optimum combination must be found as shown in Figure 3.1. Since different configurations are required by different applications there are many standards to fulfil the requirements of these applications.

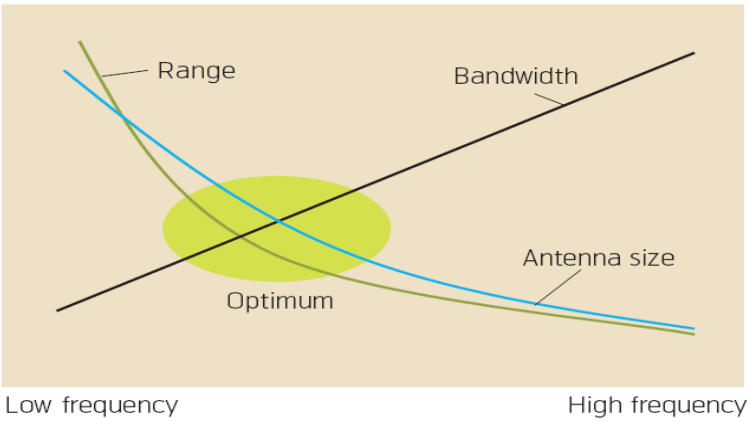

Figure 3.1: Relationship between Range, Bandwidth and Antenna Size [18]

Brief investigating of the wireless network standards defined by IEEE will help to get a better understanding of 802.15.4 as well as giving a chance to compare it with the other standards.

3.2 Wireless Network Standards

Main activities on wireless networks are going on by IEEE 802 standardization group [19]. The activities can be split into three groups:

- Wireless Personal Area Network (802.15.xx)
- Wireless Local Area Network (802.11xx)
- Wireless Metropolitan Area Network (802.16x)

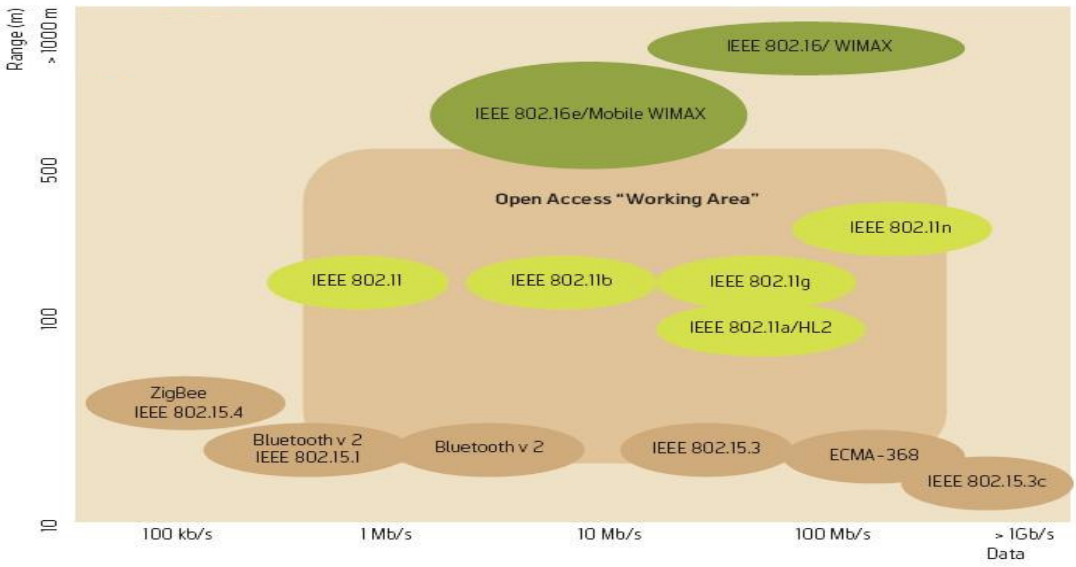

Figure 3.2: Data and Range Comparison of the Wireless Network Standards [18]

Data and range comparison of the standards in these three groups can be found in the Figure 3.2 with different colour tones.

3.2.1 802.15 Base Standard

The IEEE 802.15 Working Group for WPAN [20] has developed and released several standards for WPANs utilizing different frequency bands and providing different data rates. The main working groups in the 802.15 area are:

802.15.1: Bluetooth version 1.1, standard has been published in 2002. It offers up to 1 Mb/s data rate and a range up to approximately 100 m, depending on the power class. It operates in the unlicensed industrial, scientific and medical (ISM) band at 2.4 to 2.485 GHz

802.15.3: High Data Rate Wireless PAN (HR-WPAN) providing data rates from 11 to 55 Mb/s in the 2.4 – 2.485 GHz ISM band. The standard was approved in 2003

802.15.3a: Enhancement of 802.15.3 with multimedia and very high data rate extensions up o 480Mbit/s based on a ultra wide band (UWB) technique operating from 3.1 – 10.6 GHz**.** Unfortunately project is stopped since a consensus has not been reached.

802.15.3c: Millimetre Wave Alternative in Physical Layer, which will operate in the 57 – 64 GHz bands offering data rates of up to 2 – 3 Gb/s. It is currently under work and it is supposed to finish in second half of 2008.

802.15.4: Low data rate WPAN systems, standard has been published in 2003 with data rates between 20kbit/s to 250kbit/s. The standard is intended for the use in sensor networks with ultra long battery life time of up to 5 years.

802.15.4a: Extension of the 802.15.4 standard larger range, higher data rates up to 1Mbit/s and localization. Two optional physical layers are standardized as IEEE 802.15.4a, Low Rate Alternative PHY; one based on UWB in the 3.1 – 10.6 GHz band and one direct sequence chirp based in the ISM 2.4 GHz band.

3.2.2 802.11 Base Standard

The IEEE 802.11 family of standards has been successful for home and enterprise wireless local access. Especially 802.11g is widely accepted among the producers. The 802.11 group is working on the standardization of a WLAN system. Different versions of the standard have been published, increasing the data rate up to 54Mbit/s.

802.11: The original IEEE 802.11 standard covered the physical and MAC-layers at 2.4 GHz with supported data rates of 1 and 2 Mb/s. Frequency Hopping Spread Spectrum (FHSS) technique is used as the basic air interface.

802.11b: It specifies a higher rate physical layer in the same band of 802.11 and available since 1999, operates in unlicensed 2.4 GHz band using Direct Sequence Spread Spectrum (DSSS), and supports average data rates of 1, 2, 5.5, and 11 Mbps.

802.11g: IEEE 802.11g is a physical layer extension to enhance the performance of the 802.11b compatible networks in 2003 by increasing the data rate up to 54 MBit/s.

802.11a: High data rate version in the 5GHz band with up to 54 MBit/s data rate. Modulation scheme is Orthogonal Frequency Division Multiplexing (OFDM) with a flexible carrier modulation up to 64QAM. Published in 1999.

802.11n: Since 2003 Task Group *n* with the aim of standardizing a new physical and MAC layer for both 2.4 GHz and 5 GHz networks with the ability of providing 108 Mbps data rate is working.

3.2.3 802.16 Base Standard

The IEEE 802.16 working group has the task to standardize a wireless metropolitan area network (MAN). The Base 802.16 standard is compatible with frequencies from 10 GHz to 66 GHz. After 802.16a, 802.16d, and 802.16e extensions have been added. Two important extensions are:

802.16-2004: Published standard with different option for the modulation and the used frequency bands. The standard has been published in 2004 and contains the original 802.16 standard from 2002 and the 802.16a and 802.16d extensions of the standard. TDD and FDD duplexing modes are supported. The main attention is on the OFDM mode of the standard in the 3.5GHz licensed bands.

802.16e: Extension of the 802.16-2004 standard including a mobility component. Different proposal based on OFDM are under discussion. Here the main focus is the inclusion of an enhanced channel coding scheme and mobility handling. Under discussion are Turbo-Codes and LDPC codes. The direction is clearly towards LDPC codes since TC codes are already an option in the existing standard.

Table 3.1: Comparison of the Wireless Standards [18]

Standard	**Bit rates Offered on the Physical Layer**	**Frequency Band(s)**	**Available Channels at Bandwidth**	**Transmitter Power levels**	**Typical Range**	**Main Applications**
IEEE 802.15.1 / Bluetooth v1.1	1Mb/s (v 1.1, 1.2) 1-3Mb/s (v2.0 + ERD)	ISM 2.4 GHz	79 ch at 1MHz	100mW (Class 1 radios) 2.5mW (Class 2 radios) 1mW (Class 3 radios)	100m 10m 1m	Connecting Devices Cable Replacement WPAN
IEEE 802.15.3	11 – 55 Mb/s	ISM 2.4 GHz	5 ch at 11Mhz	< 100 mW EIRP	~10 m	Portable consumer digital imaging and multimedia applications
IEEE 802.15.3c	2 – 3 Gb/s	57 – 64 GHz			A few meters	High speed internet access, streaming content download, real time streaming and wireless data bus for cable replacement
IEEE 802.15.4 (Zigbee)	20, 40, 250 kb/s	868.3 MHz (USA) 915 MHz (USA) ISM 2.4 GHz	1 ch at 2 MHz 10 ch at 2 MHz 16 ch at 5 MHz	< 100 mW EIRP (2.4 GHz)	10 – 100 m	Home automation, Remote monitoring and control
IEEE 15.4a:		3.1 – 10.6 GHz (UWB band, USA) ISM 2.4 GHz		- 41.3 dBm/MHz (0.074 μW/MHz) < 100 mW EIRP	1 – 10 m 10 – 500 m	Communication and high precision ranging / location capability (1m accuracy and better), high aggregate throughput, and ultra low power
IEEE 802.11	1, 2 Mb/s	ISM 2.4 GHz	13 ch at 22 MHz 3 non-overlapping	< 100 mW EIRP (Europe)	10 – 500 m	WLAN and hotspot
IEEE 802.11b	5.5, 11 Mb/s	ISM 2.4 GHz	13 ch at 22 MHz 3 non-overlapping	< 100 mW EIRP (Europe)	10 – 300 m	WLAN and hotspot
IEEE 802.11g	6 – 54 Mb/s	ISM 2.4 GHz	13 ch at 22 MHz 3 non-overlapping	< 100 mW EIRP (Europe)	10 – 250 m	WLAN and hotspot
IEEE 802.11a	6 – 54 Mb/s	5 GHz bands	126 ch at 20 MHz 12 non-overlapping	< 200 mW / 1 W (Europe)	10 – 200 m	WLAN and hotspot
IEEE 802.11n	Up to 200 Mb/s	ISM 2.4 GHz 5 GHz bands		< 100 mW EIRP (Europe)	10 – 500 m	WLAN and hotspot
IEEE 802.16e	240 Mb/s	< 6 GHz, licensed and unlicensed bands	In the 3.5 GHz band: 10 ch at 20 MHz 160 ch at 1.25 MHz Or any combination	Depends on frequency band	300 m – a few kilo-meters	WMAN, Mobile Broadband

3.2.4 Comparison of the Standards

Various standards have been proposed as an appropriate solution for different necessities. Many other wireless standards from other organizations (For instance; ECMA-386 from European Computer Manufacturers Association (ECMA), ETSI HiperLAN/2 from European Telecommunications Standard Institute (ETSI)) also exist. In Table 3.1 detailed information is provided for mentioned standards.

After investigating the wireless standards from IEEE, in next part unlicensed Industrial, Scientific and Medical (ISM) band and coexistence in this band will be explained briefly.

3.3 Interference in the 2.4 GHz Industrial, Scientific and Medical (ISM) Band

The industrial, scientific and medical (ISM) radio bands were originally reserved internationally for industrial, scientific and medical purposes rather than communications. However in recent years these bands have also been shared with license-free error-tolerant communication applications such as wireless LANs and cordless phones. Also microwave ovens which are operating at 2.45 GHz use this band.

One of the major problems with parallel activity of different systems in one frequency band is the interference and performance degradation. From this point of view performance of 802.15.4 devices are closely related with the other devices in the environment that are using the ISM band. Other two widely used wireless technologies in the ISM Band are 802.11 (Wi-Fi) and 802.15.1 (Bluetooth).

The impact of IEEE 802.11 stations with high traffic rate against IEEE 802.15.4 stations may be extremely critical if the same carrier frequencies are selected while the impact of Bluetooth is much less due to its frequency hopping scheme in the ISM band [21]. In the same study it is shown that the impact of a microwave oven was negligible when the distance from the oven was above 1 m. Since 802.11 stations are most probable interference sources for 802.15.4 devices, frequency coexistence should be carefully examined

3.3.1 802.15.4 and 802.11 Coexistence in the ISM Band

IEEE 802.15.4 Channels 1 to 11 are reserved for the lower frequency bands. The centre frequency of each band can be found as:

$$f_C = 2404 + 5(k-11) \;\; f_C = 2404 + 5(k-11) \qquad \textbf{(3.1)}$$

where k is the channel number in the 2.4 GHz band. In Figure 3.3 spectra of channels in 2.4GHz are shown.

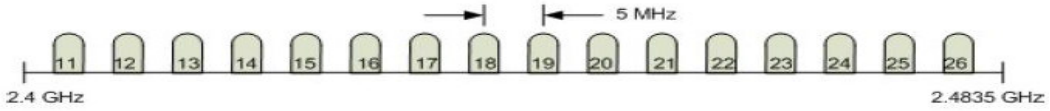

Figure 3.3: 802.15.4 Operating Channels in the 2.4GHz Band

Widely used IEEE 802.11g stations may interfere 802.15.4 stations. Each frequency channel in the 802.11g standard spans for 22 MHz, and there are 11 such channels from which 3 channels are non-overlapping. As illustrated in Figure 3.3, the IEEE 802.15.4 standard employs frequency channels of 2 MHz bandwidth which is one eleventh of the IEEE 802.11g stations. Figure 3.4 shows an illustration of the frequency spectrum relationship of IEEE 802.15.4 and IEEE 802.11g. Successful coexistence can be achieved if an IEEE 802.11g network is planned to use the non-overlapping channels 1, 6 and 11 and IEEE 802.15.4 network is planned to use channels 15, 20, 25 and 26.

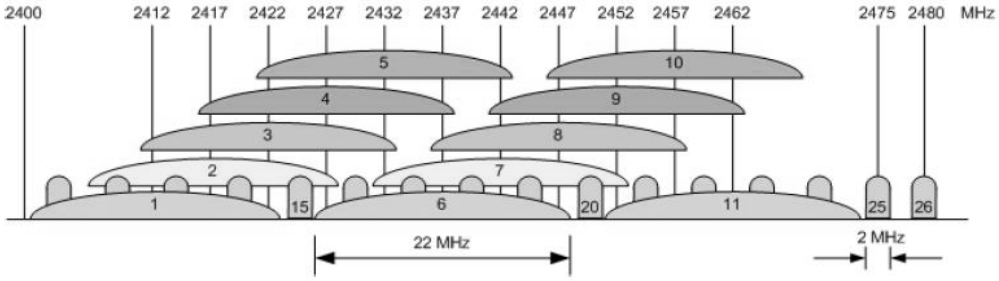

Figure 3.4: Spectrum Relationship between 802.15.4 and 802.11g

In the third part until this point, fundamental information about relationship between frequency, range, bandwidth, and antenna size; explanation of 802 bases and interference in ISM band constituted the preliminaries before going into further details in the 802.15.4 standard. Now it is time to purely concentrate on 802.15.4.

3.4 802.15.4 Overview

The main purpose of 802.15.4 standard is to provide ultra low complexity, ultra low power consumption and low cost wireless networking solution in low data rate wireless networks. The IEEE 802.15.4 protocol specifies the Medium Access Control (MAC) sublayer and physical layer for Low-Rate Wireless Personal Area Networks (LR-WPAN). Even thought this standard was not specifically developed for wireless sensor networks (WSN), it is intended to be suitable for them since sensor networks can be built up from LR-WPANs.

The IEEE 802.15.4 protocol is deeply connected with the ZigBee protocol. The ZibBee Alliance has been working together with IEEE (task group 4) in order to specify a full protocol stack for low cost, low power, low data rate wireless communications. The model of the ZigBee/IEEE 802.15.4 protocol architecture is shown in Figure 3.5. Since ZigBee Specifications are beyond the scope of this thesis project we will only investigate the Medium Access Control (MAC) sublayer and Physical Layer in the next parts. Detailed information about IEEE 802.15.4 Standard can be found at [22-23].

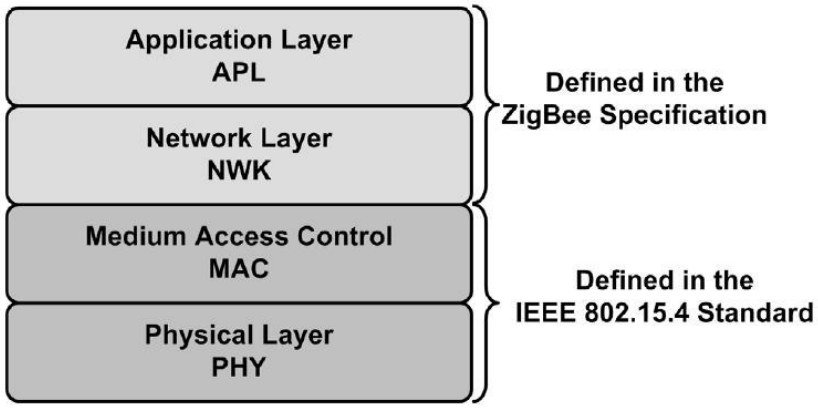

Figure 3.5: ZigBee/ IEEE 820.15.4 protocol stack architecture

3.4.1 Network Devices

Two types of devices are described in a LR-WPAN by the IEEE 802.15.4 standard:

3.4.1.1 Full Function Device (FFD)

The FFD can operate in three modes:

- **Personal Area Network (PAN) Coordinator:** Coordinates its own network, to which other devices may be associated. A LR-WPAN must include at least

one FFD acting as a PAN coordinator that provides global synchronization services to the network.

- **Coordinator:** Provides synchronization services by transmitting beacons, but it does not create its own network. Coordinator must be associated to a PAN coordinator.
- **Simple device:** A device which does not have the previously described two functionalities.

3.4.1.2 Reduced Function Device (RFD)

RFD is a device operating in minimal resources and memory capacity. An RFD is intended for applications that are extremely simple, such as a light switch or a passive infrared sensor; they do not have the need to send large amounts of data and may only associate with a single FFD at a time.

3.4.2 Network Topologies

There are two topologies defined in the IEEE 802.15.4 Standard:

3.4.2.1 Star Topology

In the star topology shown in Figure 3.6 the communication is established between devices and a single central controller, called the PAN coordinator. Each device (FFD or RFD) joining the network and willing to communicate with other devices must send its data to the PAN coordinator. Since PAN coordinator has power-consuming tasks in the star topology, the IEEE 802.15.4 standard suggests that the PAN coordinator be mains powered while other devices are more likely to be battery powered. Recommended applications in the standard are home automation, personal computer peripherals, toys and games. Consequently, the star topology seems to be not adequate for traditional wireless sensor networks, due to all sensor nodes are supposed to be battery-powered.

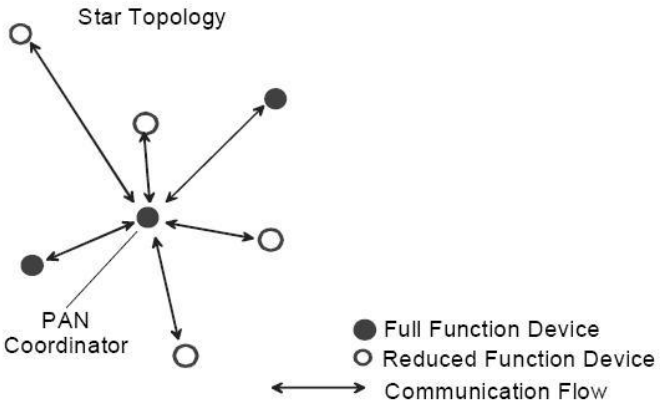

Figure 3.6: Star topology example

3.4.2.2 Peer-to-Peer Topology

The peer-to-peer topology shown in Figure 3.7 differs from the star topology in that any device can communicate with any other device as long as they are in range of one another. It also has a PAN coordinator since all LR-WPANs must have a PAN coordinator; however PAN coordinator in peer-to-peer topology doesn't have centralized tasks such in case of the star topology. Peer-to-peer topology allows more complex network formations to be implemented in contrast to star topology. Applications such as industrial control and monitoring, wireless sensor networks, asset and inventory tracking, intelligent agriculture, and security would benefit from such a network topology.

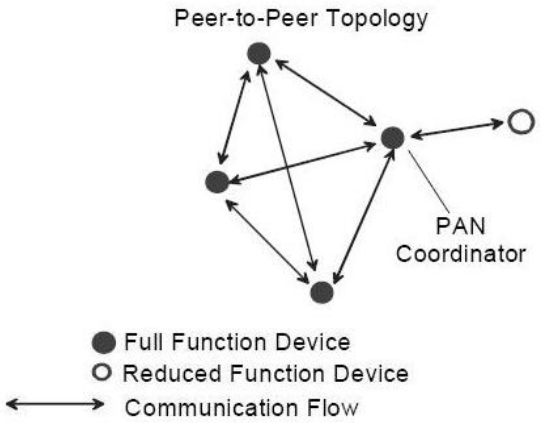

Figure 3.7: Peer-to-peer topology example

3.5 802.15.4 Physical Layer

The physical layer is responsible for data transmission and reception. The tasks of physical layer are [23]:

- Activation and deactivation of the radio transceiver
- Energy Detection (ED) within the current channel
- Link Quality Indication (LQI) for received packets
- Clear Channel Assessment (CCA) for Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA)
- Channel frequency selection
- Data transmission and reception

The IEEE 802.15.4 offers three operational frequency bands: 2.4 GHz, 915 MHz and 868 MHz. There is a single channel between 868 and 868.6 MHz, 10 channels between 902 and 928 MHz, and 16 channels between 2.4 and 2.4835 GHz. Technical details of these bands are shown in Table 3.2.

Table 3.2: Frequency bands and data rates in IEEE 802.15.4

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	2400–2483.5	2000	O-QPSK	250	62.5	16-ary Orthogonal

3.6 802.15.4 MAC Layer

The MAC sub-layer of the IEEE 802.15.4 protocol provides an interface between the physical layer and the higher layer protocols. The MAC sublayer handles all access to the physical radio channel and is responsible for the following tasks [23]:

- Generating network beacons if the device is a coordinator.
- Synchronizing to the beacons.
- Supporting Personal Area Network (PAN) association and disassociation.

- Supporting device security.
- Employing the Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) mechanism for channel access.
- Handling and maintaining the (Guaranteed Time Slot) GTS mechanism.
- Providing a reliable link between two peer Medium Access Control (MAC) entities.

3.6.1 Operational Modes

There are two operation modes described in the IEEE 802.15.4 Standard:

3.6.1.1 Beacon - Enabled Mode

When the beacon-enabled mode is selected, coordinator uses a periodic structure (Superframe Structure) to manage communication between devices. The Superframe is bounded by frame beacons as shown in Figure 3.8. The format of this structure is determined by the coordinator and transmitted to other devices inside the beacon frame. The superframe is divided into 16 equally sized slots.

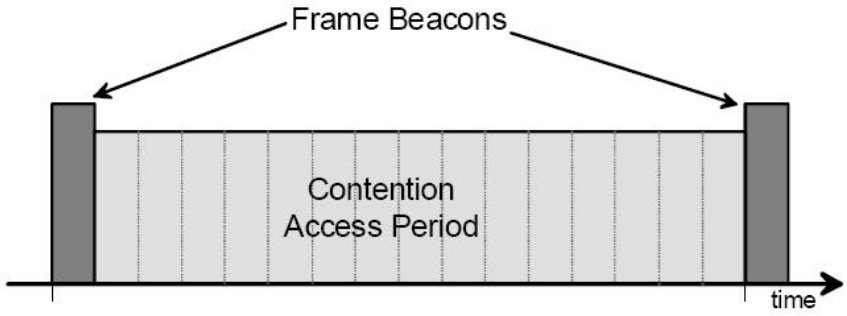

Figure 3.8: Superframe Structure without CFP

In order to offer some Quality of Service (QoS), a Contention-Free Period (CFP) is defined by the Standard. The CFP consists of Guaranteed Time Slots (GTSs) that may be allocated by the PAN coordinator to the devices that need low-latency or specific data bandwidth. The CFP is a part of the superframe and starts immediately after the CAP, as shown is Figure 3.9

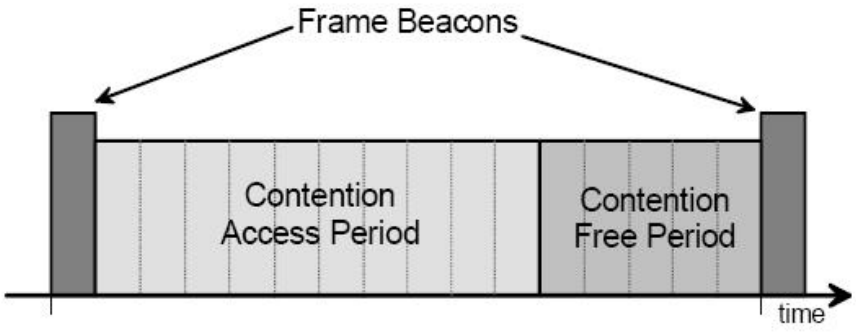

Figure 3.9: Superframe Structure with CFP

A device wishing to communicate in CAP must compete with other devices using a slotted Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) mechanism described in the IEEE 802.15.4 Standard [23]. All transmissions must be finished before the end of the CAP. A device having a GTS must ensure that its transmission be in the allocated GTS.

3.6.1.2 Non Beacon – Enabled Mode

In non beacon-enabled mode, there is no use of a superframe structure. Medium access control is provided by an unslotted CSMA/CA mechanism [23].

Summary of operational modes is shown in Figure 3.10.

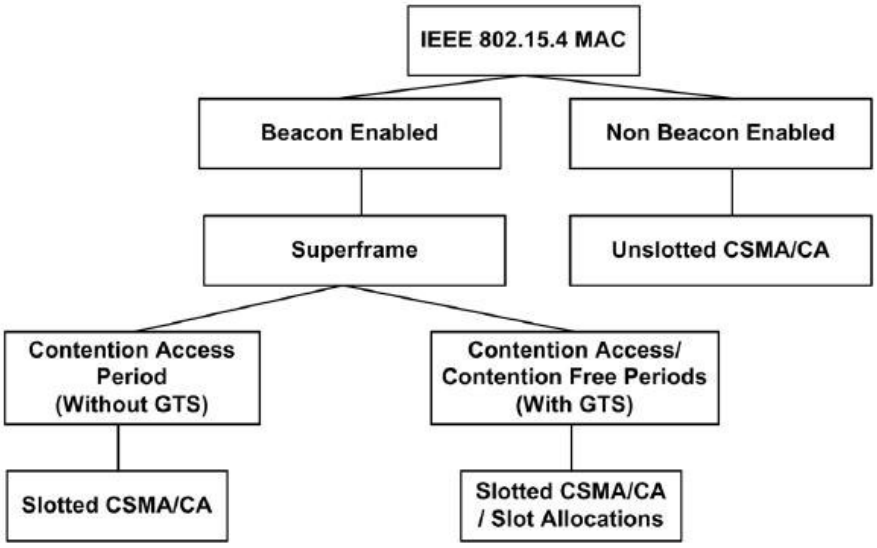

Figure 3.10: IEEE 802.15.4 operational modes

3.6.2 Superframe Structure

The superframe in a beacon interval may have an active and inactive period as shown in Figure 3.11. The coordinator interacts with its PAN during the active period, and enters in a low power mode in the inactive period

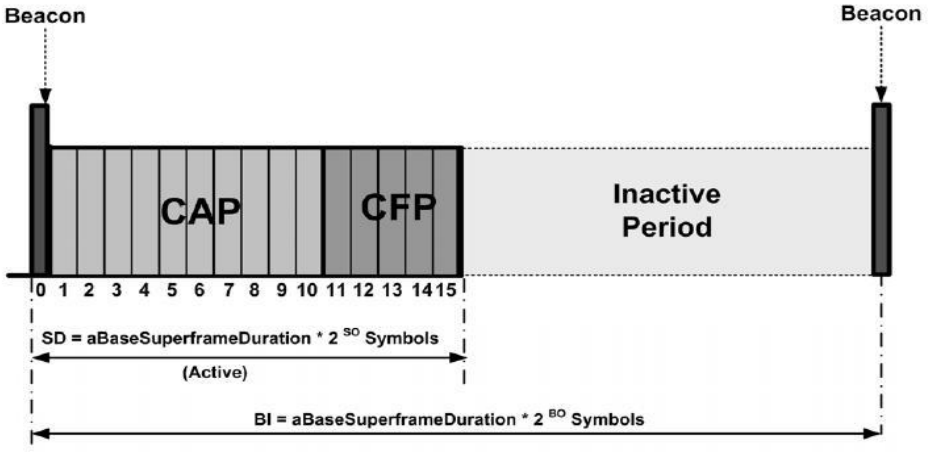

Figure 3.11: Structure of a superframe

The structure of a superframe is defined by two parameters:

- ***macBeaconOrder* (BO):** describes the time interval between the beacon frames. The value of the *macBeaconOrder* (BO) and the Beacon Interval (BI) are related as follows:

$$BI = aBaseSuperframeDuration * 2^{BO} \ symbols \quad \textbf{(3.2)}$$

- ***macSuperframeOrder* (SO):** describes the length of the active portion of the beacon interval. The value of the *macSuperframeOrder* (SO) and the Superframe Duration (SD) are related as follows:

$$SD = aBaseSuperframeDuration * 2^{SO} \ symbols \quad \textbf{(3.3)}$$

If $SO = BO \Rightarrow SD = BI$ and then the superframe is always active. $0 \le SO \le BO \le 14$ should be satisfied for beacon enable modes. $SO = 15$ means superframe will not be active.

3.6.3 Data Transfer Model

Three different data transfer type exist in IEEE 802.15.4. These are; Data transfer to a coordinator from the device; data transfer from a coordinator to the device; data transfer between two peer devices.

3.6.3.1 Data Transfer to a Coordinator

In a beacon-enabled network the device first listens for the network beacon. When a beacon is received, the device synchronizes itself to the superframe structure defined in this beacon. The device transmits its data frame, using slotted CSMA-CA. If it is requested from the device, the coordinator acknowledges the successful reception of the data by transmitting an acknowledgment frame. This sequence is shown in Figure 3.12.

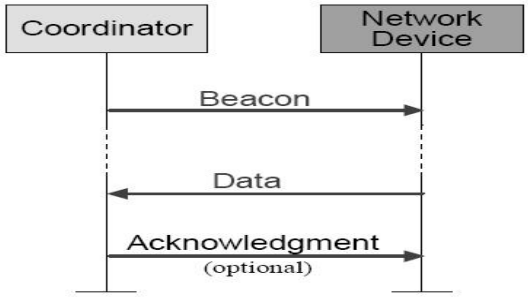

Figure 3.12: Communication to a coordinator in a beacon-enabled network

In a nonbeacon-enabled network the device transmits its data frame, using unslotted CSMA-CA. If it is requested from the device, the coordinator acknowledges the successful reception of the data by transmitting an acknowledgment frame. This sequence is shown in Figure 3.13.

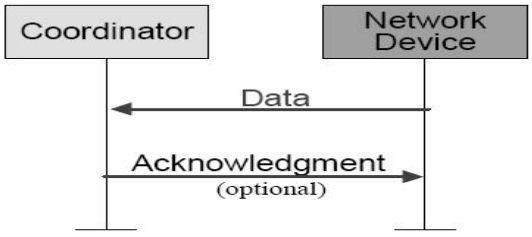

Figure 3.13: Communication to a coordinator in a nonbeacon-enabled network

3.6.3.2 Data Transfer from Coordinator

In a beacon-enabled network the coordinator indicates in the network beacon that the data message is pending for the device. When the device receives this beacon, transmits a MAC command requesting the data, using slotted CSMA-CA. If it is

requested from device, the coordinator acknowledges the successful reception of the data request by transmitting an acknowledgment frame. The pending data frame is then sent using slotted CSMA-CA by the coordinator. The device acknowledges the successful reception of the data by transmitting an acknowledgment frame. After receiving the acknowledgement, the message is removed from the list of pending messages in the beacon frame by the coordinator. This sequence is shown in Figure 3.14.

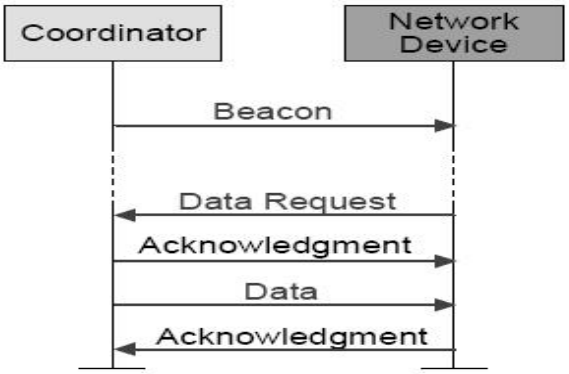

Figure 3.14: Communication from a coordinator in a beacon-enabled network

In a nonbeacon-enabled network to transfer data to the device, the coordinator stores the data for the device until request of data. The device makes contact by transmitting a request of data, using unslotted CSMA-CA, to its coordinator at an upper layer application defined time. The coordinator sends an acknowledgement frame to indicate the reception of the data request. The coordinator transmits the data frame, using unslotted CSMA-CA. The device sends an acknowledgement frame to indicate the reception of the data. This sequence is shown in Figure 3.15.

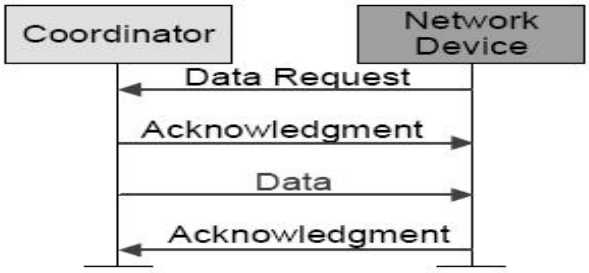

Figure 3.15: Communication from a coordinator in a nonbeacon-enabled network

3.6.3.3 Peer-to-Peer Data Transfer

In a peer-to-peer PAN, every device may communicate with every other device in its radio sphere. The device can simply transmit its data using unslotted CSMA-CA. But in order to communicate without any problem in a peer-to-peer network intelligent algorithms are needed on the upper layer which is beyond the scope of IEEE 802.15.4 Standard.

3.6.4 Frame Structure

In order to achieve robust transmission on a noisy channel, IEEE 802.15.4 Standard defines four frame structures; named as beacon frame, data frame, acknowledgement frame, and MAC command frame. All communication between the coordinators and devices are done by using these frames. Before examining the frames, common fields in four frame structures are observed. These common fields can be seen in four different frame structures at figures between 3.16 - 3.19

3.6.4.1 Common Fields

Frame Control: 16-bits long field that contains information about the frame type and related with other control flags (Security Enabled, Frame Pending, Acknowledgment Request, Intra-PAN …).

Sequence Number: It is an 8-bit field that contains a unique number for each transmitted frame.

Destination PAN Identifier: It is a 16-bit field that contains the PAN identifier of the recipient of the frame.

Destination Address: It is either a 16-bit or 64-bit field (depending on the used addressing mode) that contains the address of the recipient of the frame.

Source PAN Identifier: It is a 16-bit field that contains the PAN identifier of the sender of the frame.

Source Address: It is either a 16-bit or 64-bit field (depending on the used addressing mode) that contains the address of the sender of the frame.

The MAC Footer (MFR): It contains the Frame Check Sequence (FCS) field. The FCS is a 16 bit Cyclic Redundancy Check (CRC) sequence.

3.6.4.2 Beacon Frame

It is responsible for the synchronization in the LR-WPAN when the beacon enabled mode is used. Figure 3.16 demonstrates the fields in it meanwhile, showing the length of each field. Beacon Frame contains four distinct fields; Superframe Specification, GTS, Pending Addresses, and Beacon Payload.

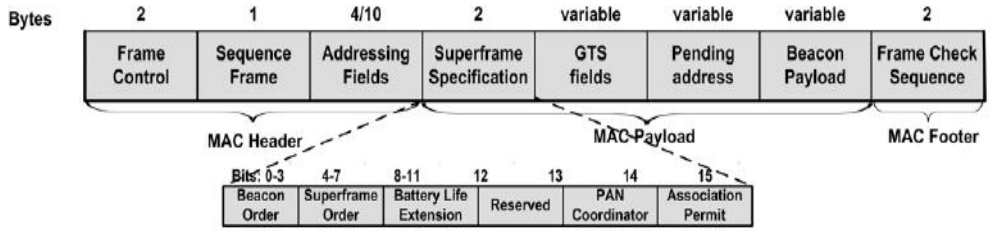

Figure 3.16: Schematic view of the beacon frame

Superframe Specification: It is a 16-bit field that specifies parameters such as Beacon Order, Superframe Order, Battery Life Extension, PAN coordinator, Association Permit.

GTS field: It is a variable size field and contains information about being allocated GTSs.

Pending Address: it is a variable size field and contains addresses of devices that have messages currently pending on the coordinator

Beacon Payload: it is an optional field reserved for upper layer to transmit data in the beacon frame.

3.6.4.3 Data Frame

It is the frame that is used to send data across the upper layers. It contains one distinct field; Data Payload.

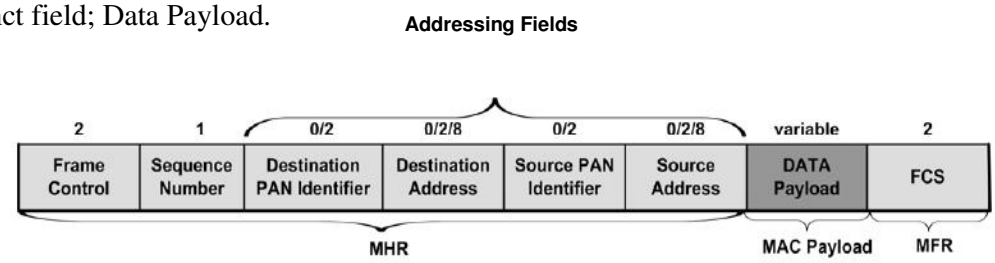

Figure 3.17: Schematic view of the data frame

DATA Payload Field: It is the field reserved for upper layer to transmit data.

3.6.4.4 Acknowledgement Frame

It is the frame that is used for acknowledgement.

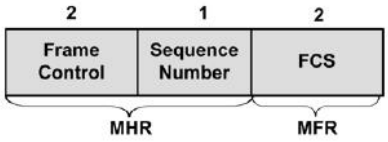

Figure 3.18: Schematic view of the acknowledgment frame

3.6.4.5 MAC Command Frame

It is the frame that is used to send MAC level commands (Data Request, Purge Request, Associate Request, GTS Request …). It contains one distinct field; MAC Payload.

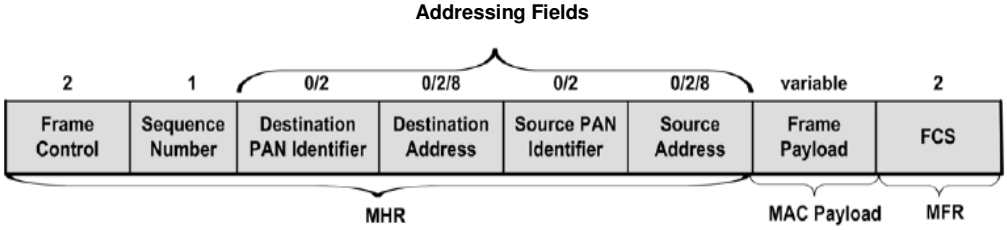

Figure 3.19: Schematic view of the MAC command frame

MAC Payload: it contains information specific to individual frame types (Data Request Frame, Purge Request Frame, Associate Request Frame, GTS Request Frame …).

3.6.5 Guaranteed Time Slot (GTS) Management

3.6.5.1 Definition of the GTS:

Some portion of the superframe can be reserved exclusively to a device on the PAN. This reserved portion is called Guaranteed Time Slot (GTS). The GTS lets the corresponding device to access the wireless channel without any competition with the other devices. Its main purpose in superframe structure is to allow low-latency or constant bandwidth to specific applications.

All GTS slots should be in the Contention Free Period (CFP). Only PAN coordinator allocates GTS slots according to the allocation request comes from the devices. It

must be used only for communications between the PAN coordinator and a device. A device to which a GTS has been allocated can also transmit during the CAP. A single GTS may extend over one or more superframe slots. The PAN coordinator may allocate up to seven GTSs at the same time.

3.6.5.2 GTS Allocation:

A device must send a GTS allocation request command to its PAN coordinator in order to allocate GTS. This message indicates the desired number of GTSs and the direction of it. The PAN coordinator sends an acknowledgement frame to confirm the receipt of GTS allocation request. The allocation of GTS is made in a FIFO order base by the PAN coordinator so, sending the GTS allocation request is not sufficient condition to allocate a GTS. It may not be allocated since there might not be a free slot in CFP. The PAN coordinator makes allocation decision within *aGTSDescPersistenceTime* (default values is 4) superframes. The requesting device keeps tracks of beacon frames for at most *aGTSDescPersistenceTime* superframes. If there is no information in the beacon frame about the GTS allocation within the time, the GTS request is considered to have failed. If the GTS was successfully allocated, the PAN coordinator sets the *start slot* field in the GTS fields of beacon frame to the superframe slot at which the GTS begins, and the *GTS Length* field in the GTS fields of beacon frame to the length of the GTS requested for the device.

3.6.5.3 GTS Usage:

When the MAC sublayer is instructed to send data using the GTS, the MAC sublayer determines if it has a valid GTS successfully allocated before. If there is, before starting transmission in GTS, each device must ensure that the time required for the data transmission; acknowledgment (if requested) and the Interframe Spacing (IFS) exist in the allocated GTS. Unacknowledged and acknowledged transmissions in GTS are shown in Figure 3.20. If GTS is sufficient in time, the data is sent. In the case of not receiving the beacon at the beginning of a superframe, the device considers all of its GTSs are deallocated. In such a circumstance device must not use its GTS until it receives a subsequent beacon correctly.

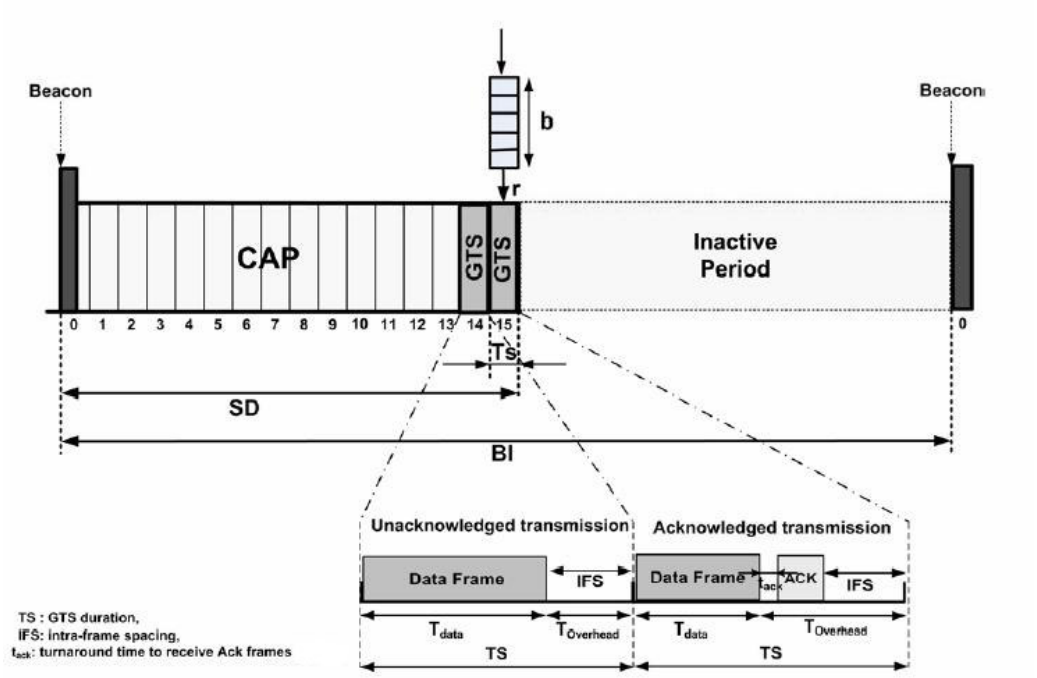

Figure 3.20: Unacknowledged and acknowledged transmissions in GTS

3.6.5.4 GTS Deallocation:

In order to request a deallocation of an existing GTS, the device sends a GTS deallocation request to its PAN coordinator. If the GTS deallocation request command is received correctly, the PAN coordinator sends an acknowledgment frame to confirm receipt. After the receipt of a GTS deallocation request command, the PAN coordinator checks that if it is a deallocation of an allocated GTS. If it is, it deallocates the GTS. If it is not, the PAN coordinator ignores the request.

3.6.5.5 GTS Reallocation:

The deallocation of a GTS may result fragmented CFP. An instance of fragmentation after a deallocation is shown in Figure 3.21. It is the PAN coordinator's responsibility to ensure that any gap occurring in the CFP are removed to maximize the length of the CAP.

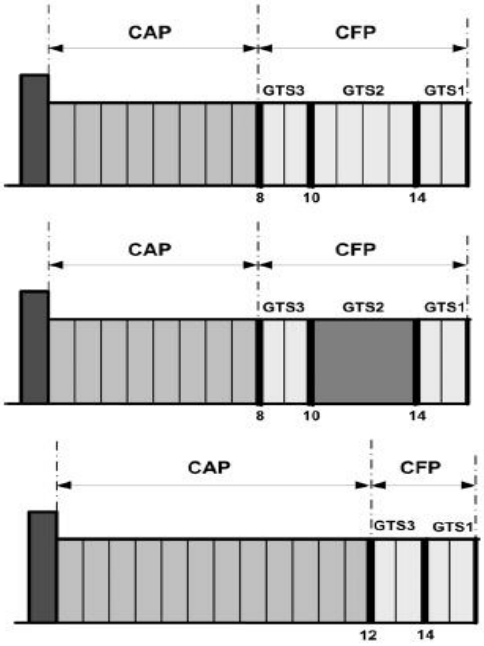

Figure 3.21: GTS Reallocation

3.6.5.6 GTS Expiration:

The PAN coordinator detects unused GTSs by the following two rules:

- For a GTS to transmit from device to the PAN coordinator, if a data frame is not received from the device in the GTS at least every $2*n$ superframes
- For a GTS to transmit from PAN coordinator to the device, if an acknowledgment frame is not received from the device at least every $2*n$ superframes

The value of n is defined as follows:

$$n = 2(8 - macBeaconOrder), \quad 0 \le macBeaconOrder \le 8 \qquad \textbf{(3.4)}$$

$$n = 1, \quad 9 \le \text{macBeaconOrder} \le 14 \qquad \textbf{(3.5)}$$

4 13192 EVOLUTION KIT (EVK) OVERVIEW

The 13192-EVK is an evolution kit from Freescale Semiconductor. The kit contains all the necessary hardware and software components to develop and demonstrate solutions based on IEEE 802.15.4 and ZigBee. Kit gives the opportunity to build wireless applications that support simple point-to-point, star or complex mesh networks.

Some Features of the kit are [24]:

- Five 2.4GHz wireless nodes based on the Freescale ZigBee-compliant platform
- 802.15.4 packet sniffer
- Onboard BDM port for MCU flash reprogramming and in-circuit hardware debugging
- RS-232 port for monitoring and Flash programming
- LEDs and switches for demonstration, monitoring and control

The 13192EVK contains two different types of boards.

- 13192-EVB
- 13192-SARD

These boards include the following on-board components:

- MC13192, 2.4GHz transceiver
- MC9S08GT60 Micro Controller Unit (MCU)
- MMA6261Q accelerometer (13192-SARD only)
- MMA1260D accelerometer (13192-SARD only)

4.1 MC9S08GT60 Microcontroller Unit

Both the 13192-SARD and 13192-EVB boards contain MC9S08GT60 Microcontroller Unit (MCU). The MC9S08GT60 is an 8-bit, low cost, low power MCU. It has 60KB of embedded flash and 4KB of RAM.

4.2 MC13192 RF Data Modem

Both the 13192-SARD and 13192-EVB boards contain the MC13192 RF data modem. The MC13192 is an 802.15.4 compliant, ZigBee-ready transceiver.

4.3 MMA1260D and MMA6261Q Acceleration Sensors

Only the 13192-SARD board contains Acceleration Sensors. Combination of MMA1260D and the MMA6261Q Acceleration Sensors provide the 13192-SARD board to act as a three axis acceleration sensor node.

4.4 13192-EVB Description

The 13192-EVB is an evaluation board based on the MC13192, 2.4GHz transceiver and the MC9S08GT60 MCU. The 13192-EVB board provides both serial and USB connectivity. It is equipped with an external SMA connector for an external antenna connection. Especially having an external antenna connector and USB connection capability makes it suitable to act as a coordinator in wireless applications. Block Diagram and Layout of 13192-EVB is shown in Figure 4.1 and Figure 4.2 respectively.

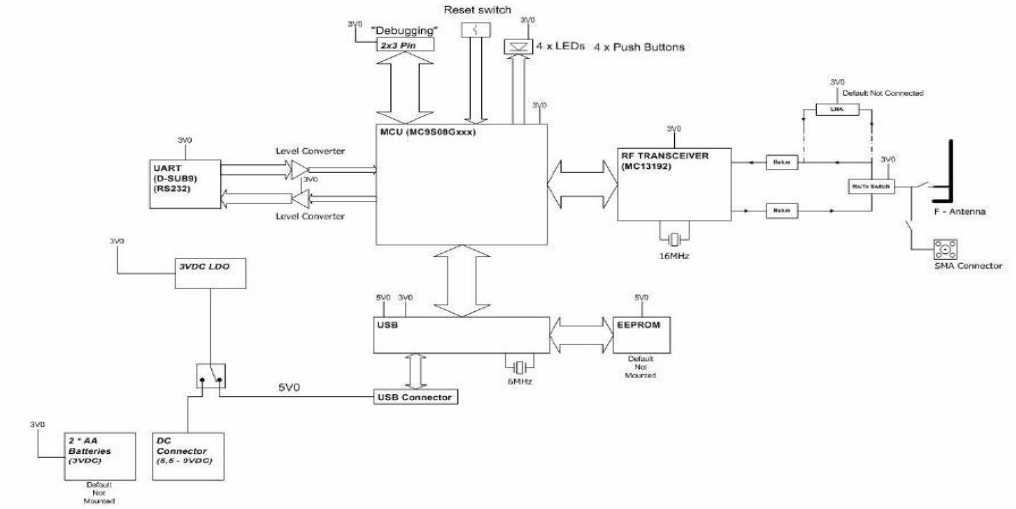

Figure 4.1: 13192 EVB Block Diagram

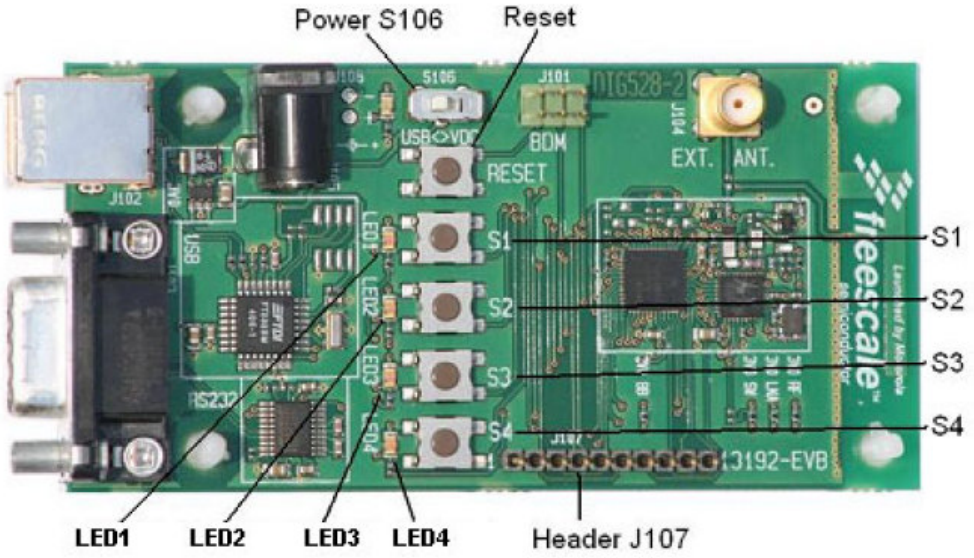

Figure 4.2: 13192-EVB Board Layout

4.5 13192-SARD Description

The 13192-SARD is a demonstration board based on the MC13192, 2.4GHz transceiver, MC9S08GT60 MCU, and the MMA6261Q and MMA1260D Acceleration Sensors. Especially having acceleration sensors makes it suitable to act as a sensor node in wireless applications. Block Diagram and Layout of 13192-SARD is shown in Figure 4.3 and Figure 4.4 respectively.

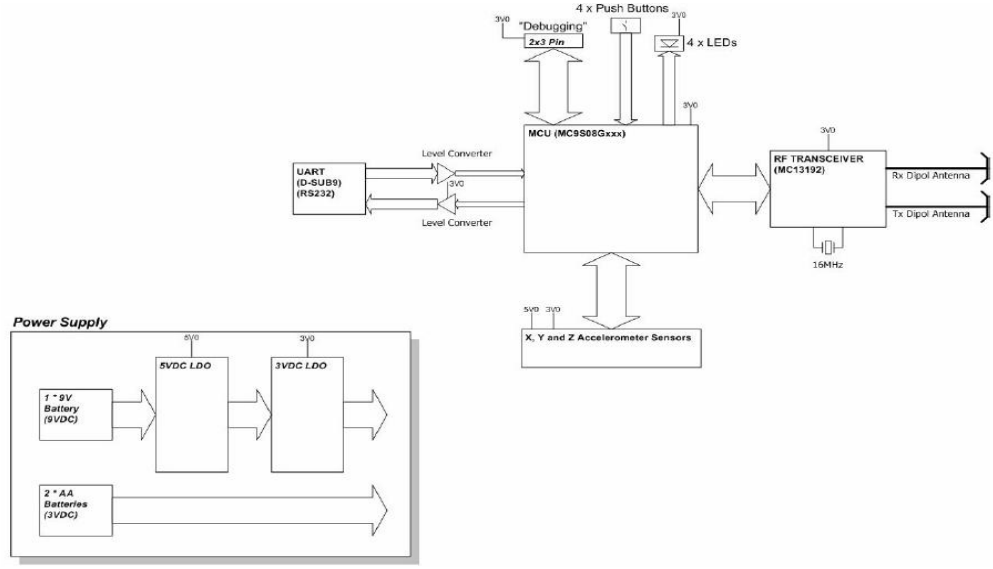

Figure 4.3: 13192-SARD Block Diagram

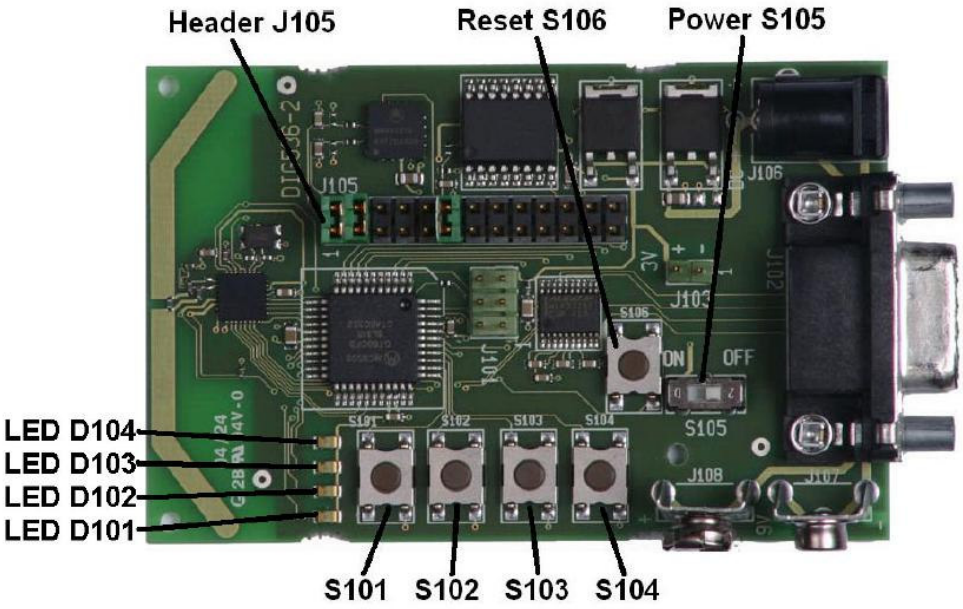

Figure 4.4: 13192-SARD Board Layout

5 GOODPUT MEASUREMENTS IN GTS

In the communication networks throughput is defined as the amount of data per unit time that is delivered over a physical or logical link. It is a criterion to compare different communication networks in capacity but when it comes how to deal with protocol overhead, the throughput is not a well-defined metric. It is typically measured as the number of bits per second that are physically delivered to the channel.

To determine the actual speed of a network or connection, the goodput measurement definition may be used. The goodput is the amount of useful information that is delivered per second to the application layer protocol. Dropped packets or packet retransmissions as well as protocol overhead are excluded.

How many bits can be delivered through one MAC layer implemented on a device to the other device's MAC layer is a question that arises in the real time applications. Throughput gives an idea about bits delivered through the physical channel but it isn't enough to measure the useful bits transferred from one device to other. As described above to find useful data transferred among the MAC layers of different devices goodput is more convenient than throughput. In the following parts, throughput and goodput performance of the 13192-EVK is compared. A MATLAB GUI and C language codes are developed with the intention of comparing these two quantities. Details of hardware setup and software can be found in the next parts.

5.1 Hardware Setup

In order to measure the goodput with GTS a 13192-EVB card in PAN coordinator mode and a 12192-SARD card in device mode but in FFD functionality are used. Details about these cards can be found in Chapter 4. Cards are connected to the PCs with RS-232 interface. The measurements are done in a laboratory environment which has 802.11 wireless networks. To achieve maximum goodput, before each measurement the interference from the 802.11 devices are checked with the

Netstumbler software. Among the non-overlapping channels of 802.11 and 802.15.4 the channel which has lowest noise power is chose. Description of coexistence in ISM band can be found in Part 3.3. The hardware setup which is used for measurements is shown in Figure 5.1.

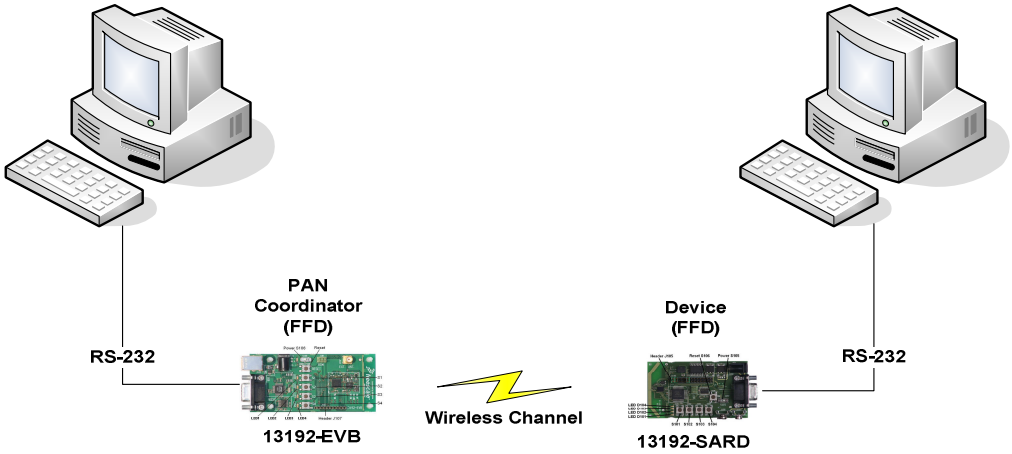

Figure 5.1: Hardware Setup

5.2 Software

The product tried to derive from these measurements is to find the amount of goodput that can be reachable using GTS in real-life applications. MAC/PHY Library Version 1.063 from Freescale is used to program devices in Network Layer. C codes using Freescale library are developed for both 13192-EVB and 13192-SARD cards. Also a MATLAB GUI is developed to reach the measurement results from a user friendly environment.

5.2.1 Freescale 802.15.4 MAC/PHY Software

Freescale 802.15.4 MAC/PHY Software [25, 26] is an IEEE 802.15.4 Standard compliant software packet. In other words the Physical and MAC layer defined in the 802.15.4 Standard is implemented in this software by Freescale. It has various types of libraries.

5.2.1.1 Libraries in Freescale 802.15.4 Software

Freescale 802.15.4 MAC Software provides different type of precompiled libraries for RFD and FFD devices to allow the best memory solution considering the functionality and physical size of used memory. In addition, it provides some derivatives for both FFD and RFD type of devices to further reduce memory

requirements. These libraries and required memory for each of them are shown in Table 5.1. During the measurements FFD library is used for both devices.

Table 5.1: MAC/PHY Software Library Functionality [26]

Device Type	Description	Typical Usage	Mac Library File Name	Code Size
FFD	Full-blown FFD. Contains all 802.15.4 features including security.	PAN coordinator, Coordinator, Router, or End-device. Includes Beacon Mode support, GTS, parameter verification and security.	802.15.4_MAC_FFD.Lib	35.4kB
FFDNB	Same as FFD but no beacon capability	PAN coordinator, Coordinator, Router, or End-device. No beacon capability is included, making this Device Type incapable of joining a beacon network. It can transmit/receive beacons for scanning. Includes security and parameter verification	802.15.4_MAC_FFDNB.Lib	24.2kB
FFDNBNS	Same as FFD but no beacon and no security capability.	PAN coordinator, Coordinator, Router, or End-device. No beacon capability is included, making this Device Type incapable of joining a beacon network. It can transmit/receive beacons for scanning. Security is not supported.	802.15.4_MAC_FFDNBNS.Lib	19.7kB
FFDNGTS	Same as FFD but no GTS capability.	PAN coordinator, Coordinator, Router, or End-device. Lacks the ability to communicate using GTS, but may participate in a Beacon Network. Includes security.	802.15.4_MAC_FFDNGTS.Lib	31.6kB
RFD	Reduced function device. Contains 802.15.4 RFD features	Operates as an End-device only and can participate in beacon networks. Includes security.	802.15.4_MAC_RFD.Lib	26.0kB
RFDNB	Same as RFD but no beacon capability.	Operates as an End-device only, and can not participate in beacon networks. Includes security	802.15.4_MAC_RFDNB.Lib	21.3kB
RFDNBNS	Same as RFD but no beacon and no security capability.	Can operate as an End-device only, and can not participate in beacon networks. Security is not supported	802.15.4_MAC_RFDNBNS.Lib	16.8kB

Since 12193-SARD card is configured as a device in the network topology, configuration that uses RFD library in 12193-SARD card also had tried, but allocation of GTS could not be possible. Most probably, Freescale have chosen not to implement GTS functionality in the RFD library, because in 802.15.4 Standard GTS functionality for RFD is optional.

5.2.1.2 Application Programming Interfaces (API)

Freescale 802.15.4 MAC/PHY Software contains three Application Programming Interfaces (APIs). These are:

- **MAC Sublayer Management Entity (MLME):** All 802.15.4 MAC commands are sent to the MAC layer with this interface. For example, applications must use this interface to send the MLME-GTS.request primitive. MLME-GTS.confirm primitive will also be received from this interface after the request. MLME interface is defined in the 802.15.4 Standard [23].
- **MAC Common Part Sublayer (MCSP):** All 802.15.4 data related primitives are sent to the MAC layer with this interface. This interface is defined in the 802.15.4 Standard [23].
- **Application Support Package (ASP):** This interface is used for application support features. For example, applications can request to enter the low power mode from this interface. ASP interface is property of Freescale.

For more detailed description of MAC primitives refer to the IEEE 802.15.4 Standard [23]. System Block Diagram related with the described interfaces is shown in Figure 5.2.

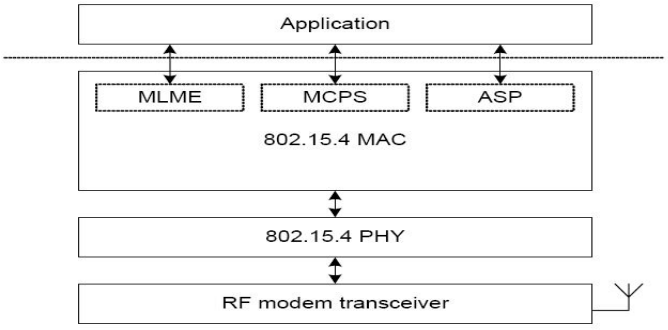

Figure 5.2: Freescale 802.15.4 Software System Block Diagram

5.2.1.3 Interfacing to the Freescale 802.15.4 MAC Software:

The interface between the application layer and the MAC layer is built on service primitives. These primitives are passed from one layer to another as messages. For instance, MCPS-DATA.request primitive which requests a data transmission to the MAC layer is sent to the NWK_MCPS_SapHandler as a message. Six SAPs exist, three for the communication to the MAC and three for the communication to the application layer. These SAPs are shown in the Figure 5.3.

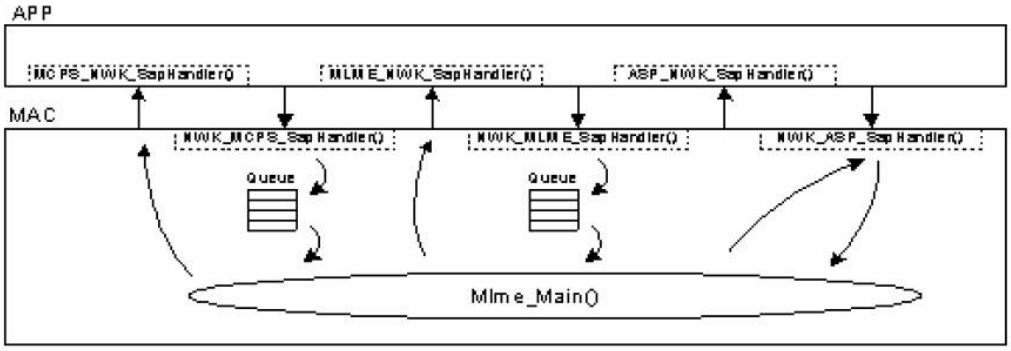

Figure 5.3: MAC Interfaces

SAPs provided by the MAC Layer:

- **NWK_MCPS_SapHandler():** Used for sending data related primitives to the MAC layer from the application layer.
- **NWK_MLME_SapHandler():** Used for sending command related primitives to the MAC layer from the application layer.
- **NWK_ASP_SapHandler():** Used for sending all ASP indications to the MAC.

SAPs must be implemented in the application layer:

- **MCPS_NWK_SapHandler():** Used for sending data related primitives to the MAC layer from the application layer.
- **MLME_NWK_SapHandler():** Used for sending command related primitives to the application layer from the MAC layer.
- **ASP_NWK_SapHandler():** Used for receiving all ASP indications from the MAC.

For allocation and of messages to send to the MAC, dequeue from the allocated messages from the buffer of MAC or sending messages to the MAC, there are three functions exist. These are shown in the Table 5.2 with their descriptions.

Table 5.2: Message Handling Functions [25]

Functions	Description
*pMsg = MSG_AllocType(type)	Allocate a message of a certain type. Used to allocate messages for one of the SAPs that are provided by MAC layer. The type parameter can be set to mlmeMessage_t, mcpsMessage_t, and aspMessage_t (See Table 5.2).
MSG_Free(*pMsg)	Frees a message that was allocated using MSG_AllocType().
Status = MSG_Send(SAP, *pMsg)	Sending a message is equal to calling a SAP function. The SAP argument can be NWK_MLME, NWK_MCPS or APP_ASP.

On the MLME, MCPS, and ASP interfaces there are six data types available. A common element of these data structures is a variable that defines the type of the message. The rest of the data structure is a union that must set up according to the message type. Data structures passed from the application layer are shown in Table 5.3 with their descriptions.

Table 5.3: Data Structures Passed From the Application Layer [25]

Data Type	Description
mlmeMessage_t	The data structure of the messages passed from the application to the MLME SAP handler.
mcpsMessage_t	The data structure of the messages passed from the application to the MCPS SAP handler.
aspMessage_t	The data structure of the messages passed from the application to ASP SAP handler.

While sending the MAC messages to the application layer similar data structures are used. Like the data structures passed from application layer a member variable contains the message type and the rest of the data structure is a union that must be interpreted according to the message type. Data structures passed from the application layer are shown in Table 5.4

Table 5.4: Data Structures Passed From the MAC [25]

Data Type	Description
nwkMessage_t	The data structure of the messages passed from the MLME to the applications MLME SAP handler.
mcpsToNwkMessage_t	The data structure of the messages passed from the MCPS to the applications MCPS SAP handler.
aspToAppMsg_t	The data structure of the messages passed from the ASP to the applications ASP SAP handler.

5.2.1.4 MAC Main Tasks in Freescale 802.15.4 Software:

Since MAC is developed asynchronous with the application layer software to make it independent from the application layer, MAC tasks must be executed through regular calls to the Mlme_Main() function which is responsible for the following duties [25] :

- Processing all primitives sent to the MLME, MCPS, and ASP SAPs.
- Answering data requests received from remote devices.
- Processing the GTS fields, pending address fields, and the beacon payload of received beacon frames.
- Automatically sending data request packets to extract pending data from remote devices if beacon mode and the PIB attribute macAutoRequest is enabled.
- Applying encryption/decryption to the MAC frames if security is enabled.

5.2.1.5 Accessing PIB Attributes in Freescale 802.15.4 Software:

The MAC PIB holds MAC attributes and variables. According to IEEE 802.15.4 Standard, MLME-SET.request and MLME-GET.request primitives, provide access to the PIB. For example to get the value of PIB attribute macBeaconTxTime in the measurements described in Part 5.5 code fragment in Appendix 5 is used.

5.2.1.6 MLME Primitives in Freescale 802.15.4 Software:

MLME primitives are sent to the NWK_MLME_SapHandler() function from application layer or to the MLME_NWK_SapHandler() function from the MAC layer. There are different kinds of MLME primitives defined in the 802.15.4 Standard.

MLME-SET.request, MLME-GET.request messages described in the code fragment for accessing macBeaconTxTime in Appendix 5 and MLME-RESET.request which is also very similar to the MLME-SET and MLME-GET requests are completed synchronously. Other messages from the application to the MLME interface are completed asynchronously. It means memory allocation of these messages should be done before sending the messages to the MLME SAP handler. Message allocation is done with `MSG_AllocType(mlmeMessage_t)` function. For example for a MLME-GTS.request code fragment in Appendix 5can be used. Type of GTS request can be an allocation or a deallocation. GTS Characteristic field shown in Table 5.4 should be filled with the appropriate value.

Table 5.5: GTS Characteristics Field Format

bits: 0-3	4	5	6-7
GTS Length	GTS Direction	Characteristic Type	Reserved

Since the MLME-GTS.request message is completed asynchronously Mlme_Main() function should be called to process this MLME primitive. GTS deallocation can be done with the same code with only difference in gtsCharacteristics parameter. gtsCharacteristics parameter should be loaded with the appropriate value from the Table 5.5. Some examples are shown in Table 5.6. In the standard naming the GTS as transmit or receive is always referenced the way from the device. For example 1 Transmit GTS means GTS used for the device to transmit packets.

Table 5.6: Some Examples for GTS Characteristics Field

GTS Characteristics Field (in Hex)	Description
0x01	Deallocation of 1 Transmit GTS
0x21	Allocation of 1 Transmit GTS
0x11	Deallocation of 1 Receive GTS
0x31	Allocation of 1 Receive GTS

5.2.1.7 MCSP Primitives

In contrast to MLME interface, the MCPS interface processes data related messages. The MCSP interface is used just like the MLME interface. The main difference between two interfaces is the message types. Allocating a MCPS message to send data using GTS is shown in the code fragment in Appendix 5. Msg_AllocType(nwkToMcpsMessage_t) function is used to send MCPS messages.

5.2.2 C Code

For both of the wireless card described in Part 4 embedded C codes are developed. Example applications from Freescale used as a template [25, 26]. Coding conventions used in these applications are followed. For example state machine architecture and variable naming used in these applications are preserved. In Figure 5.4 a simplified software flowchart of full function device is shown. In full function device received packet counter is incremented with each receive of MSCP-DATA.indication message. Since measurements are done between two MLME-GTS.request (allocation and deallocation), BeaconTxTime is sent to the PC at each allocation and deallocation to measure the time passed. See Figure 5.10. Number of received packets is also sent to the PC at each deallocation to calculate goodput.

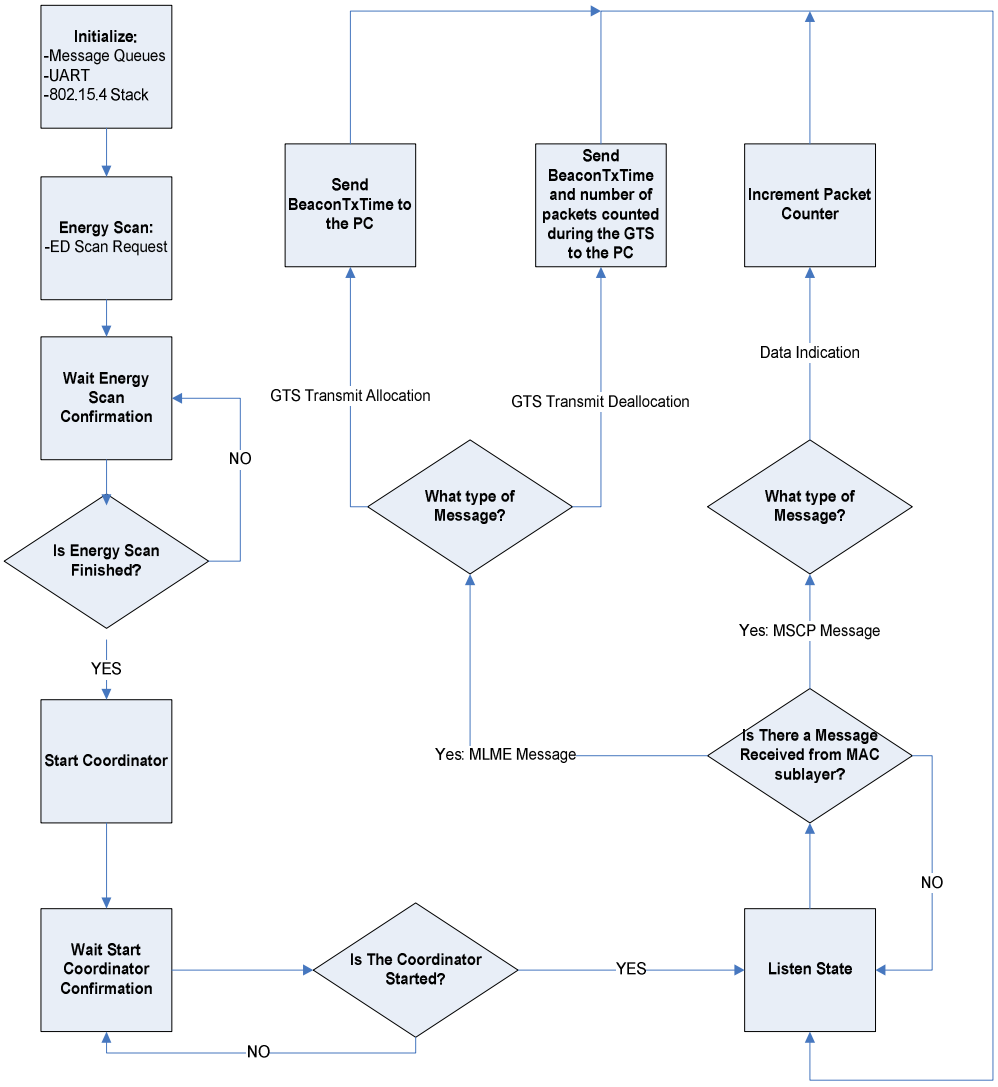

Figure 5.4: Full Function Device Simplified Software Flowchart to Measure the Goodput

In reduced function device GTS allocation is done with MLME-GTS.request primitive. After allocation, GTS slot is used to send data packets. An example code to allocate GTS can be found in Part 5.2.1. To attain the number of packet desired to be sent to the PAN coordinator, packets are counted by using the MSCP data confirmation. In each data confirmation message from the MCPS, packet counter is incremented. Reduced function device simplified software flowchart is shown in Figure 5.5.

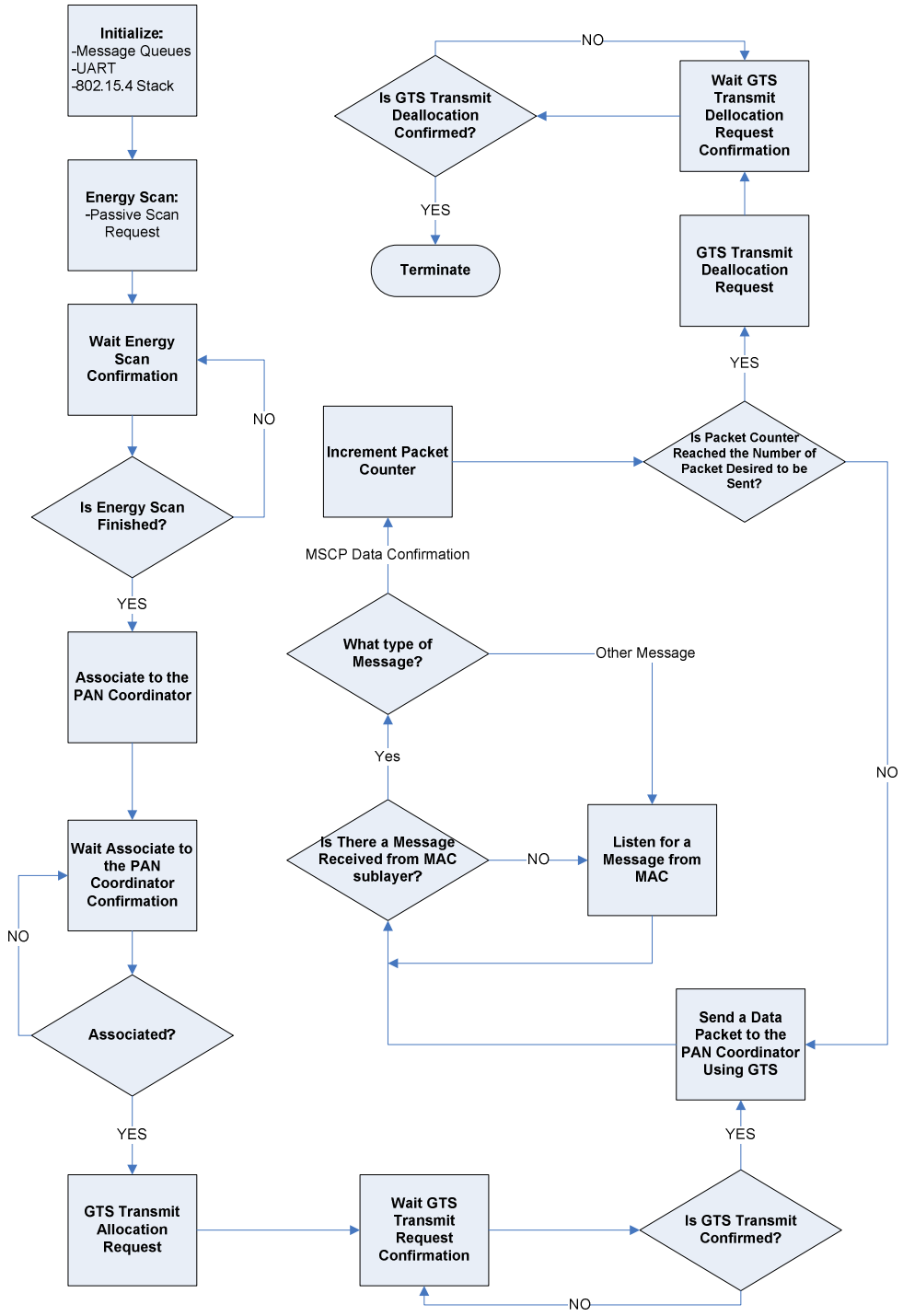

Figure 5.5: Reduced Function Device Simplified Software Flowchart to Measure the Goodput

5.2.3 MATLAB Graphical User Interface (GUI)

On the PC connected to the PAN coordinator developed MATLAB graphical user interface is used to easily access the measurement results. Interface gets the

beaconTxTime and number of received packets from the PAN coordinator to calculate the goodput. RS-232 interface is used in communication between PC and PAN coordinator. A snapshot from the GUI is shown in Figure 5.6.

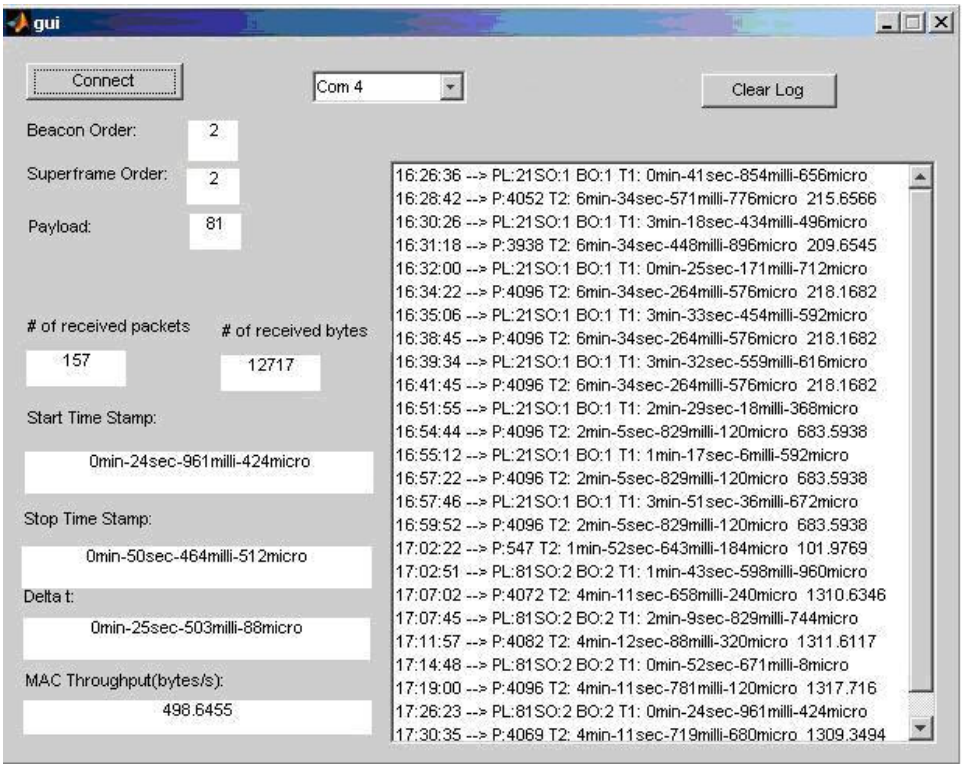

Figure 5.6: MATLAB GUI

Before allocating the GTS and starting to send packets, the device sends the payload size, beacon order and superframe order to the PAN coordinator with an indirect acknowledged data packet. Just after PAN coordinator get this packet it informs the MATLAB GUI about the payload size, beacon order and superframe order as well as beaconTxTime. Changing the superframe and beacon order by reprogramming the PAN coordinator is enough for new experimental setup. No need to reprogram the device. It automatically chooses the optimum payload size from a table according to the beacon and superframe order. Choosing the optimum payload size for fixed packet length is described in Part 5.5. MATLAB GUI converts hexadecimal values from the microcontroller unit to the decimals and calculates the goodput. After each measurement it saves the results to a text file

5.3 Maximum Throughput Evaluation

In the communication networks throughput is defined as the amount of data per unit time that is delivered over a physical or logical link. It can be formulized for GTS as in (5.1) where BI is duration of Beacon Interval, T_{GTS} is duration of GTS, and C is bit rate.

$$Th = \frac{T_{GTS}}{BI} C \qquad \textbf{(5.1)}$$

In $BO = SO$ case, since $\frac{T_{GTS}}{BI}$ is constant for a particular GTS allocation, Number of GTS allocation is the only parameter that define the throughput in a particular bit rate. For example, when $C = 250kbps$ and the number of GTS is one, throughput is always 15,625 kbps. Throughput, superframe and GTS durations for different BO values can be seen in Table 5.7

Table 5.7: GTS, Superframe Lengths and Throughput for 1 GTS

SO=BO	BI (sec.)	TGTS (sec.)	# bits in 1 GTS (kb.)	Throughput (kbps)
0	0,01536	0,00096	0,24	15,625
1	0,03072	0,00192	0,48	15,625
2	0,06144	0,00384	0,96	15,625
3	0,12288	0,00768	1,92	15,625
4	0,24576	0,01536	3,84	15,625
5	0,49152	0,03072	7,68	15,625
6	0,98304	0,06144	15,36	15,625
7	1,96608	0,12288	30,72	15,625
8	3,93216	0,24576	61,44	15,625
9	7,86432	0,49152	122,88	15,625
10	15,72864	0,98304	245,76	15,625
12	31,45728	1,96608	491,52	15,625
12	62,91456	3,93216	983,04	15,625
13	125,82912	7,86432	1966,08	15,625
14	251,65824	15,72864	3932,16	15,625

5.4 Maximum Goodput Evaluation

The goodput is the amount of useful information that is delivered per second to the application layer protocol. Dropped packets or packet retransmissions as well as protocol overhead are excluded. It may be more meaningful than throughput when it is desired to measure the performance in application layer.

Since maximum packet size for physical layer is 133 bytes without header [23], for the superframe orders higher than two, more than one packet can be fitted in one

GTS. In this point a question that arises is how the fragmentation of GTS should be done. Variable sized packets can be used to utilize GTS efficiently. But in real-time applications it may be an obstacle to reserve some microcontroller time to process variable sized packets. Also in order to have software that can be implemented easily in application layer, choosing fixed sized packets is reasonable. In all goodput calculations and measurements fixed sized packets are used.

In order to find the most efficient fragmentation of GTS interval respect to maximum goodput, expression in the (5.2) should be maximized. The set of equations through 5.3 to 5.4 defines an optimization problem which is illustrated in Figure 5.7

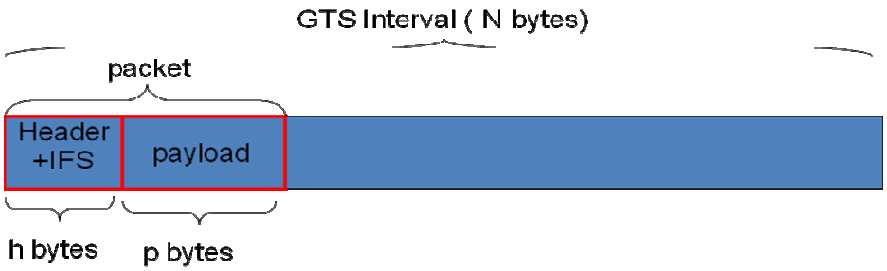

Figure 5.7: Optimization Diagram

Maximize np **(5.2)**

Subject to $nh + np \le N$ **(5.3)**

$0 \le p \le p\max \le N - h\,, \quad 1 \le n \le N/h$ **(5.4)**

h, N : constants

n: number of packets in GTS interval

$p\max$ is a standard defined maximum value for payload size and theoretically it cannot be more than $N - h$. $N - h$ defines the case when there is only one packet in GTS. Upper limit for n is N/h which represents the case of zero payload size.

In Figure 5.8 projection of the general graphical solution to the $f(np) = 0$ surface can be seen while Figure 5.9 illustrates an example case for BO=SO=5. In this case maximum value of np is obtained while n=7 and p=102.

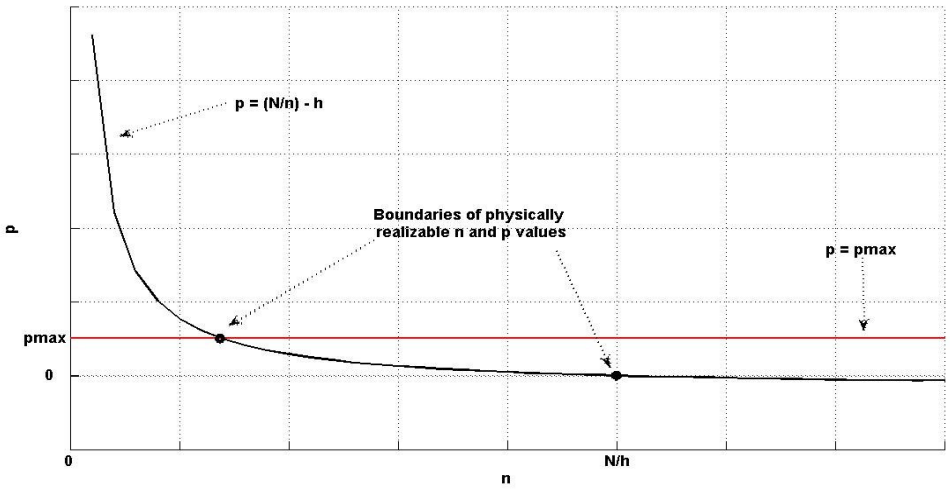

Figure 5.8: Graphical Solution to the Optimization Problem

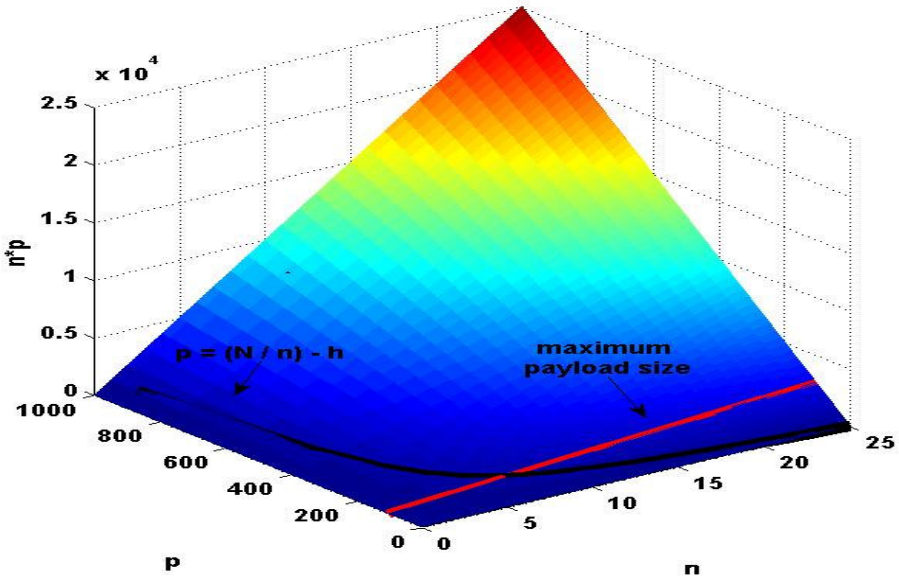

Figure 5.9: An Example case for SO=BO=5

In the 802.15.4 2006 Standard, for the star network topology, it is possible to specify only source addressing fields (size of addressing fields may be 4 bytes in short addressing mode), increasing the maximum data payload to 118 bytes. Data frame that demonstrates this case is shown in Figure 5.10. Maximum Data Payload can be found via subtracting lengths of other fields in MPDU from *aMaxPHYPacketSize*. Maximum size of Data Payload is calculated as;

$$n = 127 - (5 + 4) = 118 \; bytes \qquad \textbf{(5.5)}$$

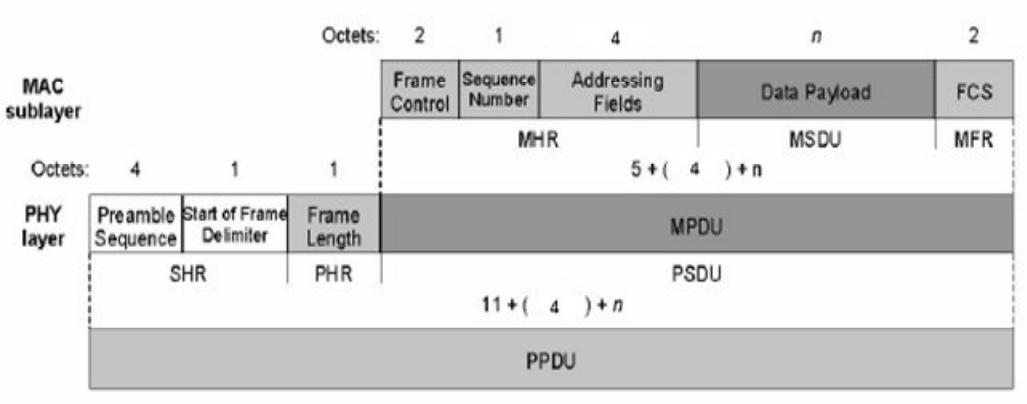

Figure 5.10: Data Frame that Demonstrates the Maximum Payload Case

All goodput and maximum payload sizes calculated by using optimization problem defined in equation set (5.2) to (5.4) for different superframe orders and different GTS allocations are given Appendix 1. Results for 1 GTS allocation are given in the Table 5.8. It should be noticed from the tables in Appendix 1 that choosing the maximum packet size is not always the most efficient way of fragmenting GTS.

Table 5.8: Maximum Payload and Goodput Values for 1 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	25	1	813.80	6.5104
2	85	1	1383.46	11.0677
3	85	2	1383.46	11.0677
4	118	3	1440.43	11.5234
5	102	7	1452.64	11.6210
6	112	13	1481.12	11.8489
7	118	25	1500.45	12.0035
8	118	50	1500.45	12.0035

5.5 Measured Maximum Goodput

By the help of calculated payloads for maximum goodput at previous part, numerous measurements are done with the setup shown in Figure 5.1 to find the maximum payload size that can be reachable experimentally. Unfortunately, software provided by Freescale does not support changing the value of aMaxFrameOverhead. It means; having less than 20 bytes sized addressing field, does not give chance to enlarge data payload size.

aMaxPHYPacketSize = **127 bytes** (Maximum PSDU Size)

aMaxFrameOverhead = **25 bytes** (Maximum Frame Overhead Size in MPDU)

Maximum Data Payload = *aMaxPHYPacketSize - aMaxFrameOverhead* **(5.6)**

Maximum Data Payload = 127 – 25 bytes = 102 bytes

As calculated in (5.6) maximum size of data payload is always 102 bytes regardless the size of addressing fields. Moreover, Freescale implementation of 802.15.4 has a field called Average Correlation, which is at the end of MPDU and it consumes one extra byte.

At each superframe order maximum payload that can be attainable without any packet loss is searched experimentally, after finding it optimization problem defined in equations 5.2 to 5.4 is solved with pmax value equal to the attained payload in the experiment. In this way it is checked that if there is another payload size which provides better fragmentation than maximum payload size that attained. If there is, it is chose in the measurements. At each measurement 4096 data packets are sent and the time kept by using beaconTxTime attribute defined at PAN Information Base (PIB). beaconTxTime value kept after the GTS allocation and before the GTS deallocation is used to find the time passed. It is also corrected by subtracting one beacon interval if the last GTS at the last beacon interval is not fully used. And the goodput is calculated with the equation (5.7);

$$Goodput = \frac{(4096 - number\ of\ data\ packets\ in\ the\ not\ fully\ used\ GTS).payload}{delta\ t} \qquad \textbf{(5.7)}$$

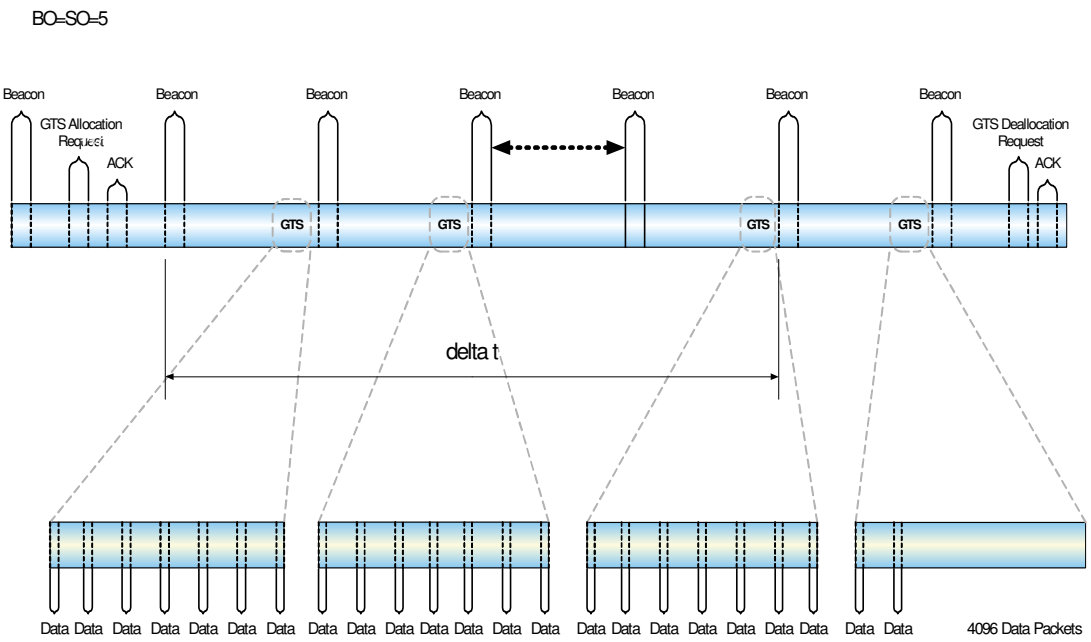

Figure 5.11: An Example Timing Diagram used in the Measurements

An example case can be seen in the Figure 5.11 for SO=BO=5. In this case 2 of the 4096 packets are in the last GTS. These 2 packets are excluded from the goodput calculation since last GTS is not fully used.

Table 5.9: Measured Goodput Values for 1 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	21	1	683.59	5.4687
2	81	1	1318.36	10.5468
3	82	2	1334.64	10.6770
4	82	4	1334.64	10.6770
5	99	7	1409.91	11.2792
6	99	14	1409.91	11.2792
7	99	28	1409.91	11.2792
8	101	55	1412.71	11.3016

Payloads and best goodputs that are measured experimentally for different orders and GTS allocations can be found in Appendix 2. Table 5.9 shows measured goodput values for 1 GTS allocation.

5.6 Results

Obtained results form Part 5.3, 5.4, and 5.5 are shown together in Appendix 3 to summarize the goodput measurements. Case in which one GTS is allocated is shown in Figure 5.12. In Figure 5.12 while moving from superframe and beacon order one to two, there is a promising improvement in the goodput. Order two is a good compromise between the throughput and latency (Lower orders mean low latency operation). Also it should be noticed that increasing the order after order five does not contribute significantly to the goodput.

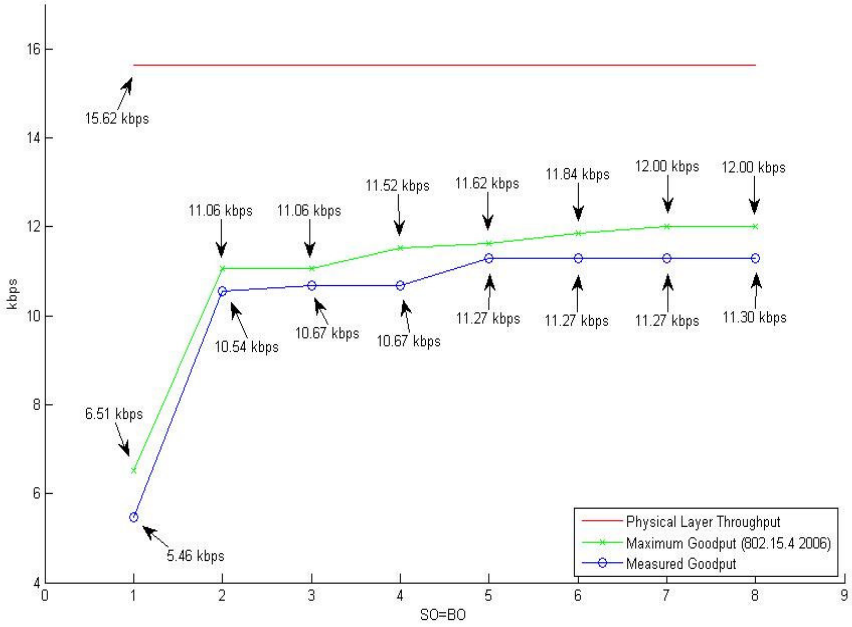

Figure 5.12: Comparison of Maximum and Measured Goodputs for 1 GTS Allocation

6 SYNCHRONOUS DATA ACQUSITION WITH SENSOR NODES

There is various time synchronization algorithms used for different scale and type of networks. For example, the Network Time Protocol (NTP) [27] is used for keeping the Internet's clocks synchronized. While measuring time of flight of sound [28] or beamforming of arrays [29], precise measure of time plays vital role. Relative synchronization among sensor node clocks is required for mentioned kind of applications. Reference-Broadcast Synchronization (RBS) [30] can be used in such synchronization necessities. With RBS algorithm, nodes send reference beacons to their neighbours using physical layer broadcasts. Receivers use arrival time of these beacons as a point of reference to compare the clocks and use this information to time stamp collected data. IEEE 802.14.5 Standard uses beacons mainly for synchronization of data and command packet transmission timings in MAC layer. IEEE 802.14.5 Standard beacon enabled mode described in Part 3.6.1 also gives opportunity to synchronize sensor nodes at application layer. With other words, beacons can be used for time stamping sensor data. Using MLME-BEACON-NOTIFY.indication primitive, analog acceleration values sampled by MC9S08GT60 microcontroller at two different 13192-SARD cards are synchronized. This synchronized data is sent to the 13192-EVB card through the allocated GTSs for low latency operation. Used star network topology is shown in Figure 6.1.

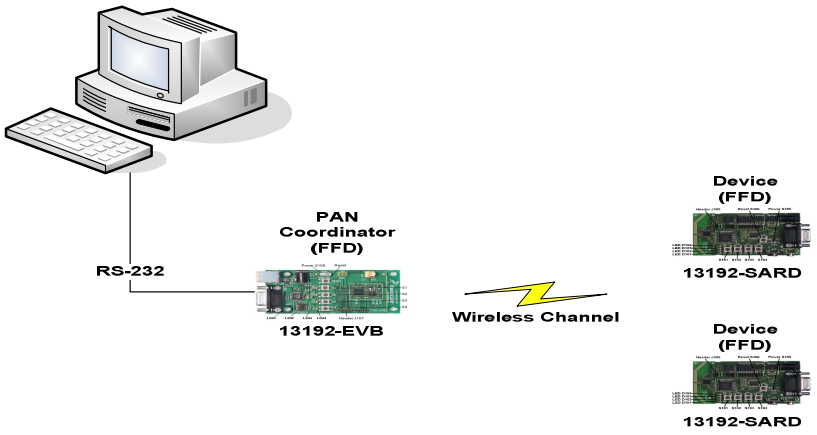

Figure 6.1: Hardware Setup Used in Data Synchronization

6.1 Choosing the Beacon Order and Number of Guaranteed Time Slot

For sensor network applications, different beacon orders and GTS allocations can be used according to the application requirements. Allocation of two GTSs for each sensor and order two for beacon and superframe is chosen.

Choosing order 2 for superframe and beacon; is a good tradeoff between latency and goodput since low orders cause low latency. This trade off significant in Figure 5.12. 2 packets with 82 bytes payload are sent from the devices in two GTSs.

6.2 Software

C codes for devices, PAN coordinator and a MATLAB Graphical User Interface to easy access the sensor data are developed.

6.2.1 Developed C Code for Device

Device's software is the most critical part of the synchronization of collected sensor data since time stamping of collected data and synchronizing sampling at each reception of beacon should be done by the devices. MLME-BEACON-NOTIFY.indication primitive generated by MAC layer at each reception of beacon from PAN coordinator is used to start the sampling of the analog sensor output. Timer of the MCU is used to generate the sampling intervals precisely. Sampled sensor data at each beacon interval is time stamped with the Beacon Sequence Number (BSN) defined in the IEEE 802.15.4 Standard [23].

6.2.1.1 Timing Diagram of Beacon Interval

Timing diagram of beacon interval is shown in Figure 6.2 where the value of two assigned to both beacon and superframe order. It can be seen that 61.44 msec long beacon interval is divided to 3.84 msec long 16 time slots. Two GTS allocations are requested by two devices and first one is at the end of superframe and second one is at the right side of it. Detailed information about GTS allocation can be found in Part 3.6.5.

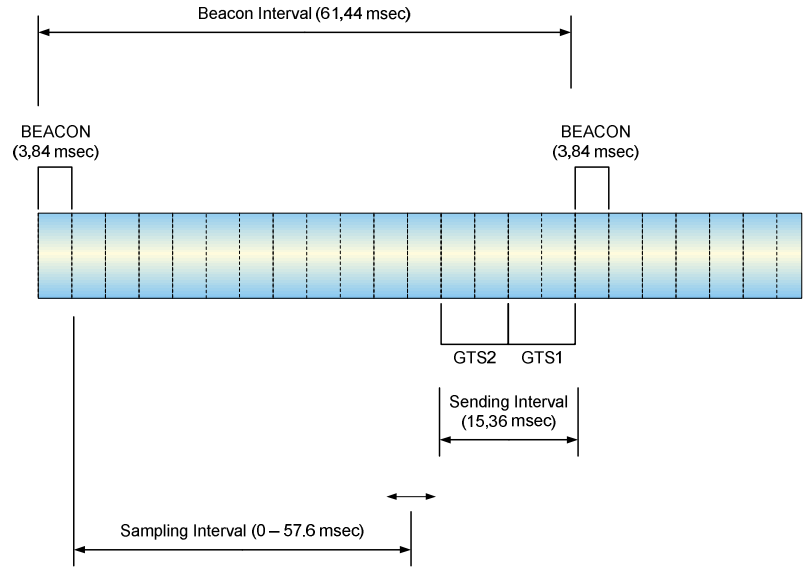

Figure 6.2: Timing Diagram of Beacon Interval

After receiving a beacon, sampling of sensors can be done in any length of interval in between the two sequential beacons. Sampling interval illustrated in the Figure 6.2 represents this stretchable interval. For easy implementation in the developed code, sampling is ended before the GTSs. In order to time stamp the collected data in the beacon interval one byte is used for each packet. Since two packets are sent, two bytes are used for time stamping in each beacon interval. It means 162 $(=2\times82-2)$ bytes of sampling data from each device can be sent in each beacon interval. In 8 bit conversion mode of analog digital converter with 162 bytes, 162 samples can be sent. In other words; no more than 162 samples can be taken from the sampling interval with configuration defined in Part 6.1. Sampling period of $200\ \mu\text{sec}$ is chose to end sampling before GTSs and give some time to the micro controller to prepare the packets, although the interval bounded by two sequential beacons can be used for sampling without losing the synchronization of sensor data.

6.2.1.2 Beacon Notify Indication

Synchronization is an obstacle in wireless sensor networks. Sensor nodes can be operational in random times or they can lost RF signal contact with the coordinator. In order to synchronize the two devices, MLME-BEACON-NOTIFY.indication is used. But in default this notification can not be received by application layer. PIB attribute macAutoRequest should be set to FALSE to get beacon notification indications from MLME. However setting macAutoRequest value to FALSE before

associating to the PAN coordinator will prevent the association since data request; which is essential for associating to PAN coordinator [23]; cannot be done automatically from a device. Association can also be done manually with macAutoRequest value set FALSE but it is chose to set macAutoRequest to FALSE after associating to the PAN coordinator and allocating the GTS. Giving effort to manually association to the PAN coordinator will not contribute to the study of synchronization of sensors. The code fragment in Appendix 5 shows how to set macAutoRequest value to FALSE.

At each reception of MLME-BEACON-NOTIFY.indication, Beacon Sequence Number (BSN) is stored in a memory location, BSN is used to time stamp the packets that will send in the same beacon interval. The code Fragment in Appendix 5 shows how to get BSN.

6.2.1.3 Timer Interrupt to Generate Sampling Frequency:

MC9S08GT60 microcontroller has two 16 bit timers. Second timer is used to generate sampling frequency. Block diagram of timer unit can be seen in Figure 6.3. Clock source can be selected among the BUSCLOCK (16 Mhz) XCLK (32Mhz) or another external clock. Selected source can be divided with the selectable prescaler.

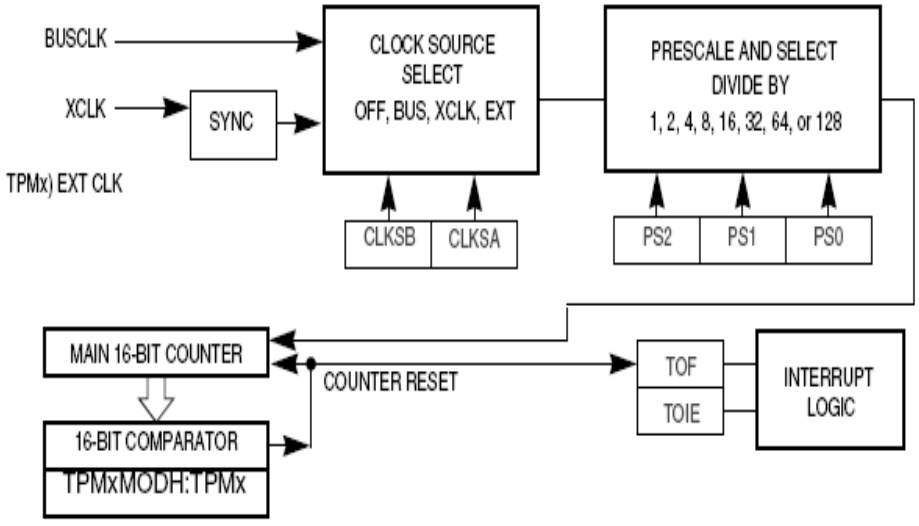

Figure 6.3: Timer Block Diagram

In order to use the timer properly three registers are necessary. These are:

- 8-bit status and control register (TPM2SC)

- 16-bit counter (TPM2CNTH,TPM2CNTL)
- 16-bit modulo register (TPM2MODH,TPM2MODL)

As the configuration parameters, clock source from BUSCLK, prescale value 4 and timer interrupt enable is chose, corresponding hexadecimal value 0x4C for configuration is assigned to the TPM2SC register in the code fragment concerned with starting timer. Prescale value 4 with using BUSCLK as a clock source means one micro second timer resolution. In other words, at each one micro second counter increments. Bit fields of TPM2SC are shown in Figure 6.4. Detailed information can be found in the datasheet of the MCU.

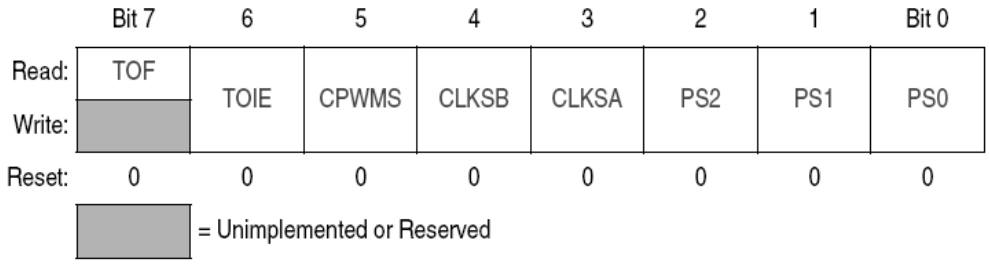

Figure 6.4: TPM2SC Register

TOF: Timer Overflow Flag

TOIE: Timer Overflow Interrupt Enable

CLKSB-CLKSA: Clock Source Sellect

PS2-PS1-PS0: Prescale Divisor Select

Since it is desired to start analog digital conversion in 200 micro second intervals, the value 200 is written to modulo registers (TPM2MODH, TPM2MODL) of the timer. Example code fragment can be found in Appendix 5.

6.2.1.4 Analog Digital Conversion to Sample the Acceleration Sensors Output

MC9S08GT60 microcontroller unit has an analog digital converter (ADC). In 13192-SARD boards three channels (AD0, AD1, AD7) of this analog digital converter is connected to the acceleration sensors mounted on board. Technical specifications of the analog digital converter are:

- 8-/10-bit resolution

- 14.0 µsec, 10-bit single conversion time at a conversion frequency of 2 MHz
- Conversion complete flag or conversion complete interrupt generation
- Analog input multiplexer for up to eight analog input channels

Block diagram of the ADC is shown in Figure 6.5

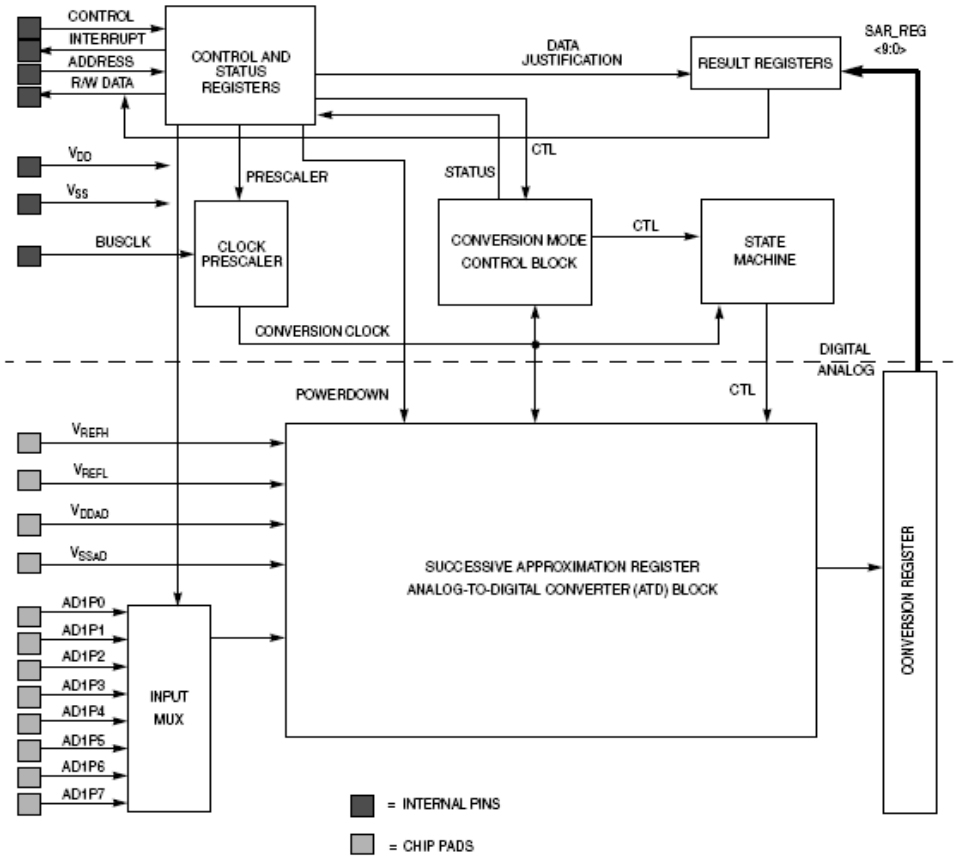

Figure 6.5: Block Diagram of Analog Digital Converter

Three registers of ADC are ATDC, ATDSC, and ATDPE important while configuring the ADC. Bit fields of these registers should be set properly.

ATDC register should be used to control the ADC. Bit fields of ATDC are shown in Figure 6.6. Detailed information can be found in the datasheet of MCU [31].

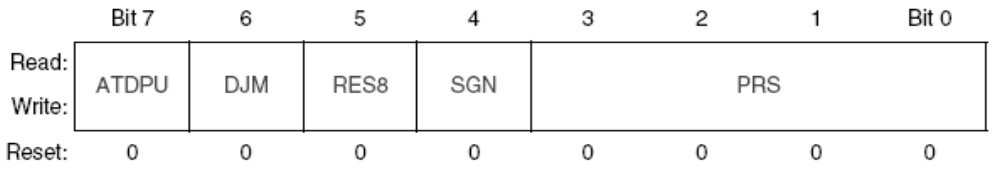

Figure 6.6: ATDC Register

ATDPU: ADC Power up

DJM: Data Justification Mode

RES8: ATD Resolution Select

SGN: Signed Result Select

PRS: Prescaler Rate Select

ATDSC register should be used to control and obtain the status of ADC. Bit fields of ATDSC are shown in Figure 6.7. Detailed information can be found in the datasheet of MCU.

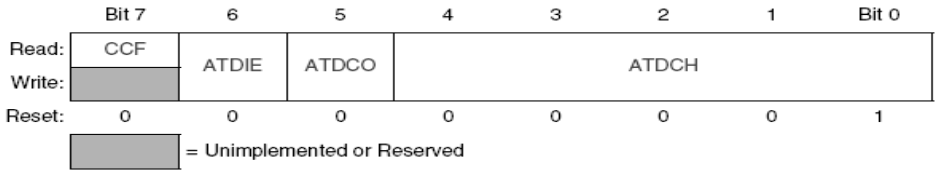

Figure 6.7: ATDSC Register

CCF: Conversion Complete Flag

ATDIE: ATD Interrupt Enabled

ATDCO: ATD Continuous Conversion

ATDCH: Analog Input Channel Select

The ATDPE register allows the pins dedicated to the ADC module to be configured for ADC usage. Bit fields of ATDPE are shown in Figure 6.7. Detailed information can be found in the datasheet of MCU. The code fragment used to utilize the ADC can be found in Appendix 5.

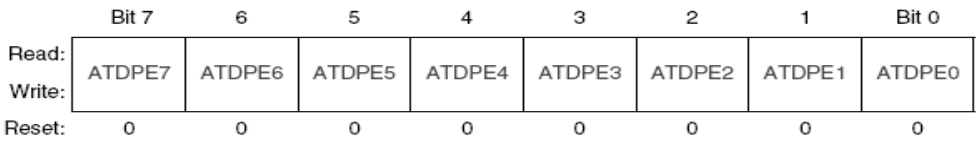

Figure 6.6: ATDPE Register

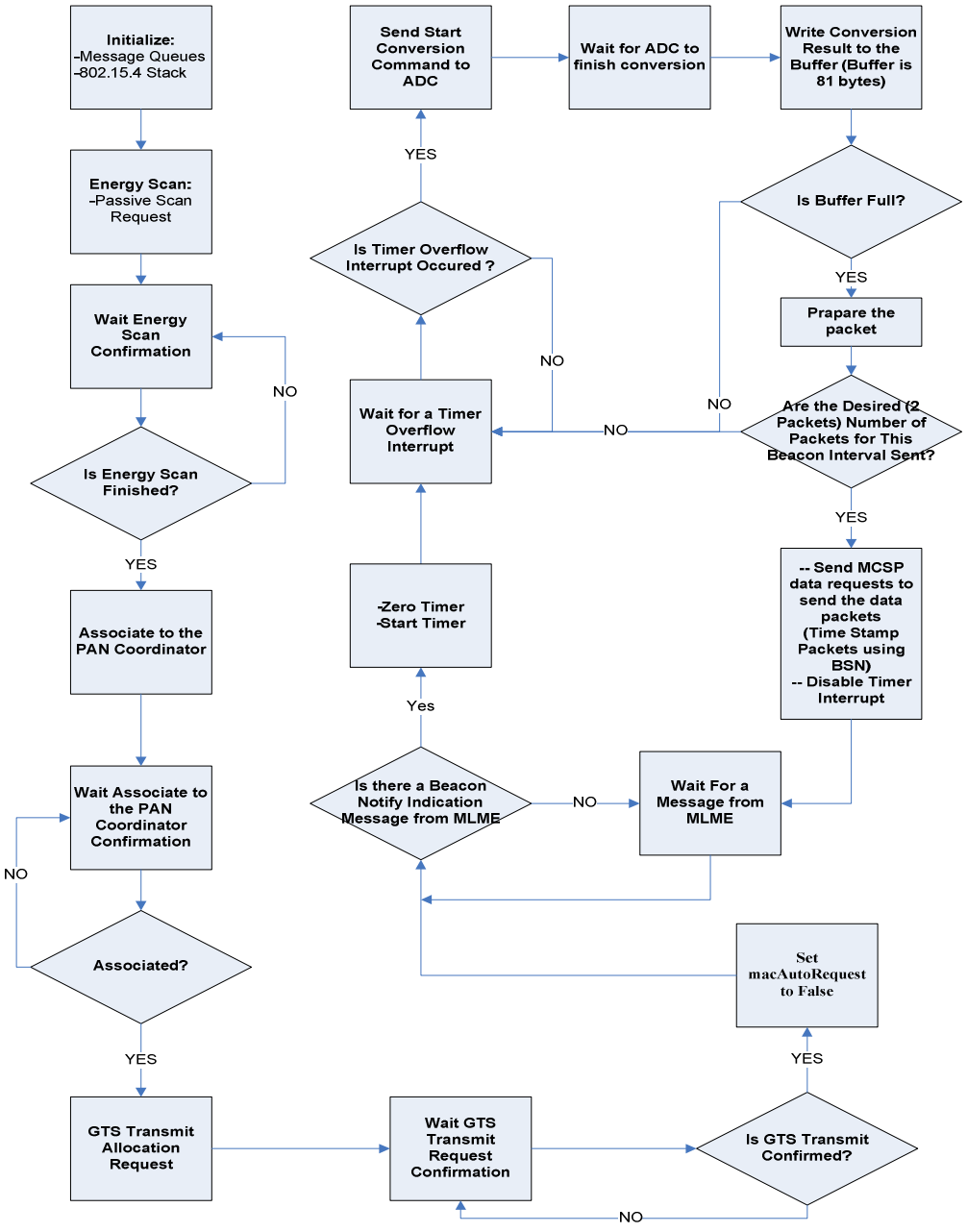

Figure 6.7: Flowchart of Reduced Function Device for Synchronization

6.2.1.5 Flow of the Software in Reduced Function Device

After associating to the PAN coordinator and allocating the GTS, devices set macAuroRequest value to FALSE in order to be able to receive MLME-BEACON-NOTIFY.indication from MAC. When the beacon notification is get from the MAC layer devices start their timer and enable interrupt for timer. Each timer overflow

triggers the devices' ADC converters to start conversion. Conversion Complete Flag (CCF) of ATDSC Register is continuously checked from the application layer until ADC finishes its conversion. Desired numbers of samples are taken and desired numbers of packets are constituted in this way. Detailed flowchart of RFD software is shown in Figure 6.7.

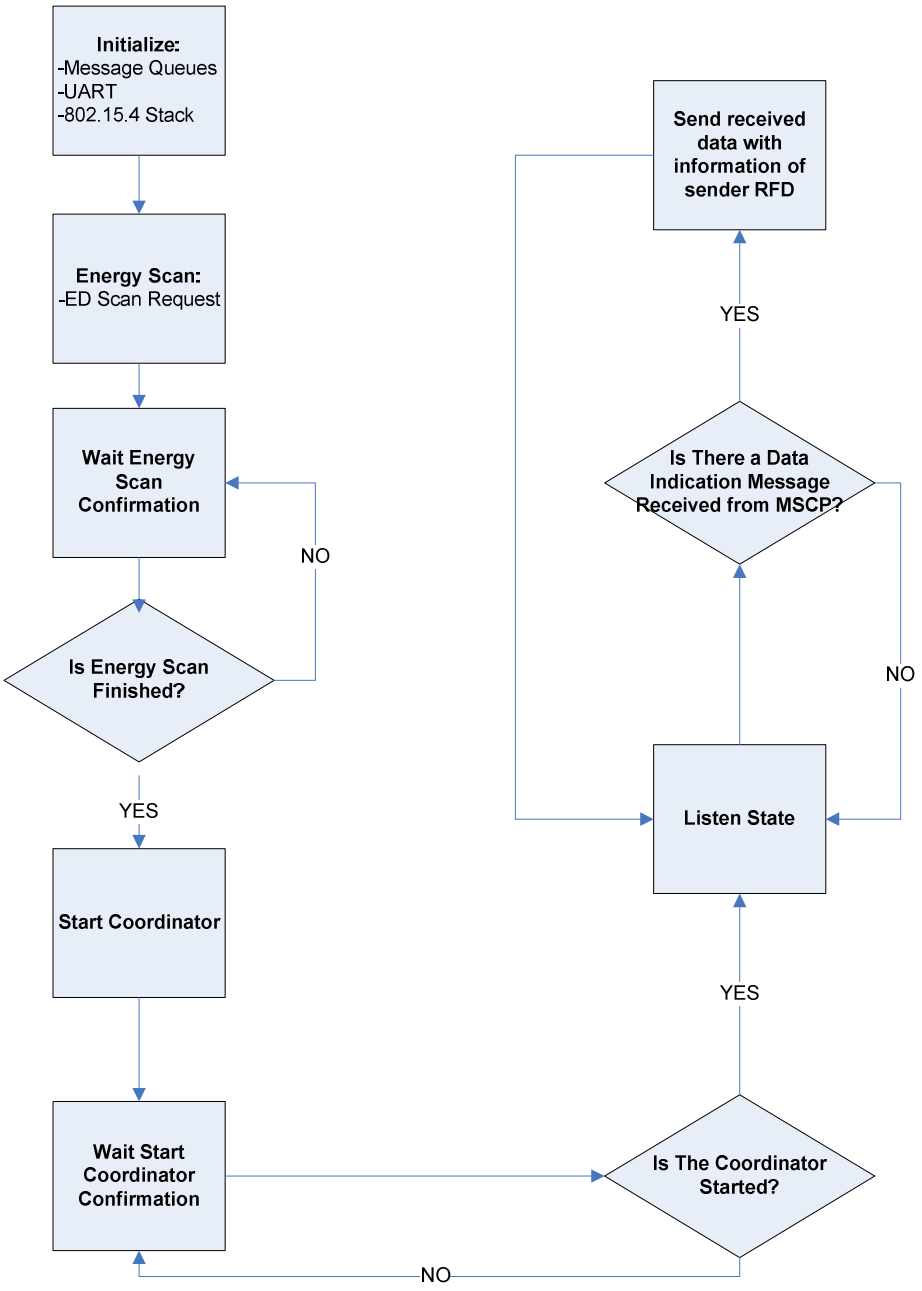

Figure 6.8: Flowchart of Full Function Device for Synchronization

6.2.2 Developed C Code for PAN Coordinator

After starting as a PAN coordinator FFD listens for the massages from devices and send the received messages to the PC through RS-232 interface. In Figure 6.8 flowchart of full function device can be seen.

6.3 Synchronization Error

With the software developed, it is observed that synchronization of acquired data between the devices is not perfect. This is the expected outcome of the used method. Error with definite boundaries is acceptable in most real life applications. In this part, characteristics of error are going to be investigated.

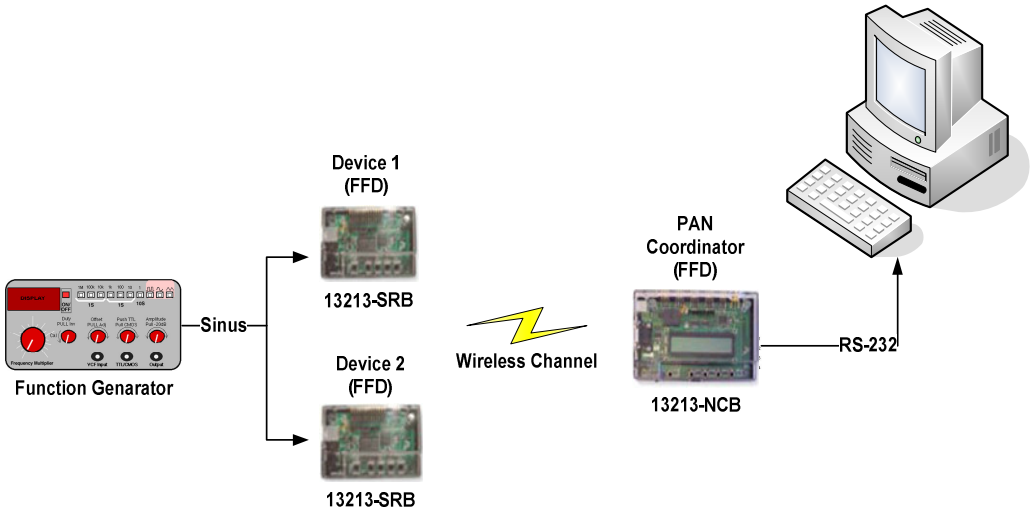

Figure 6.9: Hardware Setup Used in Tests

The data acquired from two nodes, against a sinusoidal signal sent to their inputs from a function generator, is observed as the time shifted versions of the same sinusoid. An instance of the acquired data from the devices is shown in Figure 6.10. It is observed that this time shift is random in behaviour. It is a lag or a lead having variable quantity. Used hardware setup can be seen in Figure 6.9. Synchronization error measurements are done with 1321xNSK instead of 13192EVK. These two kits have the same modem and MCU. 1321xNSK contains single in-line packaged version of MC9S08GT60 MCU and MC13192 RF Modem.

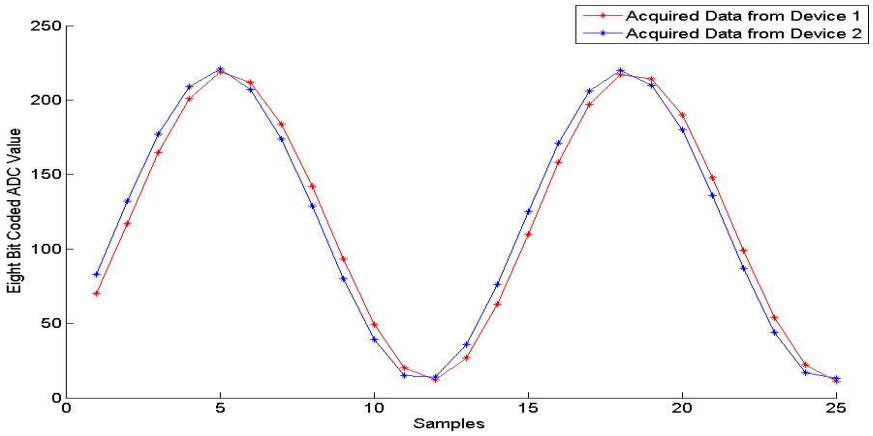

Figure 6.10: An Instance of Delay between the Devices during the Tests (1 kHz Sinus, 13.3 kHz Sampling)

6.3.1 Possible Causes of the Delay between the Devices

Since devices are synchronized with the reception of beacon from PAN coordinator and the devices are not located equidistance to the PAN coordinator, beacons reach to the antennas of devices at different times. In PAN devices should be located maximum 10 meters apart from each other for reliable communication due to the restrictions about transmitting power in the 802.15.4 Standard. Radio frequency signals can travel 10 meters roughly in $33n$ sec (light travels 3×10^8 m/s). Thus delay between devices caused by reception of beacon at different times can be maximum $33n$ sec . Delay in this order can be neglectable. Moreover, in the synchronization tests devices run very close to each other. Also this delay can not be random because of the fixed positions of the devices.

Phase difference between the clocks of devices may cause a time shift. Like the delay occurred by different travel time of beacon, this kind of delay is on the order of nanoseconds and it is not random. It can be determined at each start up of the devices with qualified equipments. With the 32 MHz clock it should be less than $31.25n$ sec ($31.25n$ sec is the period of one CPU cycle).

Software response time is another source of delay. Task scheduler running in the devices needs some time to serve again to a specific task or event. Beacon notification indication message sent from MAC layer can only be processed while task scheduler is serving to the network layer at where developed codes run. Task

scheduler has a main loop which enlarges or shrinks due to tasks waiting on the task list. With 32 MHz CPU period of task scheduler should be in the orders of microseconds. At the other hand beacons come to the devices with a period of 61.44 milliseconds with beacon order 2. We can consider periodic beacons as a clock and main loop of the task scheduler as another clock with variable frequency. This combination of two clocks crates randomness. During the tests Freescale 802.15.4 MAC/PHY 2.04 library is used.

6.3.2 Each Device Delays Randomly

When PAN coordinator transmits a beacon, each device introduces a delay before get started sampling. But this delay is introduced once just after the beacon transmission. Sampling is done in precise intervals since the timer interrupt of MCUs are used. Figure 6.11 illustrates the delays of each device while sampling a sinusoidal signal. The delay introduced by the first device is named T_1 and the delay introduced by second device is named T_2. Actually the delay observed between the devices during the tests is neither T_1 nor T_2. It is the difference between T_1 and T_2. For now on it will be named as E.

$$E = T_1 - T_2 \tag{6.1}$$

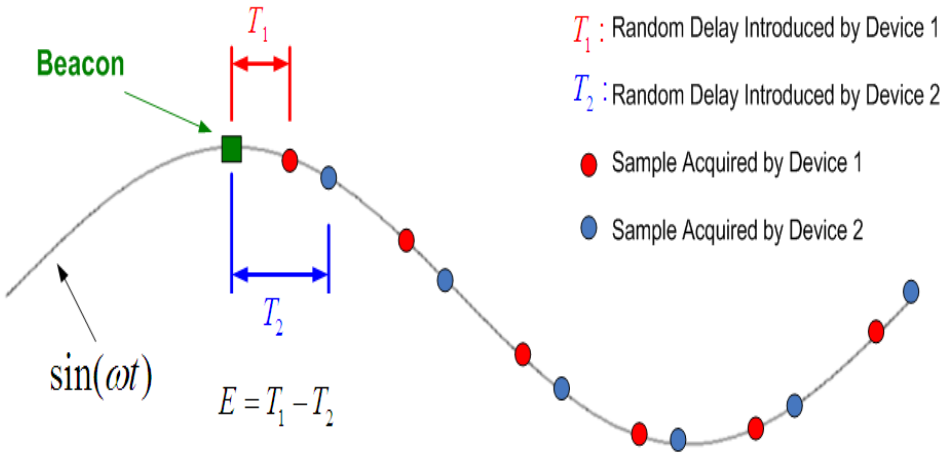

Figure 6.11: Each Device Delays Randomly

Alternative illustration of the Figure 6.11 with the same sampling points can be seen in Figure 6.12. In the Figure 6.12 acquired samples are plotted with respect to their indices. This kind of demonstration is preferable since it is in the form of Figure 6.10 which shows data acquired during the tests. N length time series that consist of the sampled values by Device 1 is named $D_1(n)$ and N length time series that consist of the sampled values by Device 2 is named $D_2(n)$.

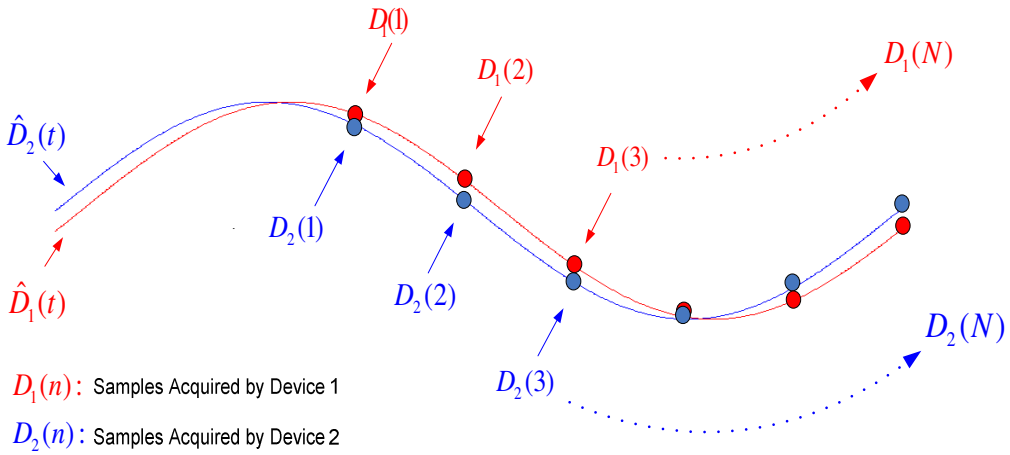

Figure 6.12: Alternative Illustration of Figure 6.11

6.3.3 Used Method to Reveal the Delay

After defining the notation that is going to be used, now it is time to find an estimation method to the value of *E*. Having $D_1(n)$ and $D_2(n)$ gives the opportunity to find estimates to the analog signals sampled by the devices. Cubic Spline Interpolation is used to construct $\hat{D}_1(t)$ and $\hat{D}_2(t)$. $\hat{D}_1(t)$ and $\hat{D}_2(t)$ are the estimates to the analog signals $D_1(t)$ and $D_2(t)$ seen by the devices.

$$D_1(n) \overset{\text{interpolation}}{\Rightarrow} \hat{D}_1(t) \tag{6.2}$$

$$D_2(n) \overset{\text{interpolation}}{\Rightarrow} \hat{D}_2(t) \tag{6.3}$$

The value of τ which will minimize the mean square error between $\hat{D}_1(t)$ and $\hat{D}_2(t-\tau)$, is calculated to find an estimation to *E*. Absolute maximum value of τ can be chosen freely but it should be grater than absolute maximum value of *E* and

less than the period of the sampled sinus minus absolute maximum value of E that can be occurred. Absolute maximum value of τ is chosen as $150\mu\sec$.

$$\hat{E} = \min_{\tau} \mathrm{E}[(\hat{D}_1(t) - \hat{D}_2(t-\tau))^2] \qquad \textbf{(6.4)}$$
$$\tau_{\max,\min} = \pm 150\mu\sec$$

6.3.4 Fitting Distributions to the Distribution of Delay

After estimating the E values for each beacon reception with the method described in Part 6.3.3, statistical distribution of $\hat{E}$ is examined. Distribution of $\hat{E}$ is tried to fit into the some commonly known distributions. These distributions are Normal, Weibull, Extreme Value, Triangle, Gamma, Uniform, Exponential, Log Normal, Rayleigh, Beta, and Exponential. Least square error between empirical and fit cumulative density functions (CDF) is used for the goodness of fit. The parameters used by fit CDFs are estimated using a standard maximum likelihood method and a least-squares error is defined as the difference between the empirical and parameter based CDF. The best fit is obtained by ranking the minimum error in the least-squares sense. In all cases with 1024 point long data sets, triangle distribution is found the best fit to the distribution of $\hat{E}$. In order to demonstrate the independency between the frequency of sampled sinusoid and $\hat{E}$ tests are done in different frequencies. Results can be found in Appendix 4. For instance, goodness of fit values to the distribution of $\hat{E}$ against a 500 Hz sinusoid at the inputs of devices can be seen in Table 6.1. Figure 6.13 shows the PDF of $\hat{E}$ and the four best ranked fits. Triangle fit is the best fit. It can be said that triangle distribution may be used to statistically model the random process E. It should be noted that in each case fitted triangle is an isosceles triangle with peak located at zero.

Table 6.1: Goodness of Fit (500 Hz Sinusoid on the Input)

Rank	Distribution	Goodness of Fit
1	Triangle	0.004
2	Normal	0.013
3	Shifted Weibull	0.025
4	Extreme Value	0.075

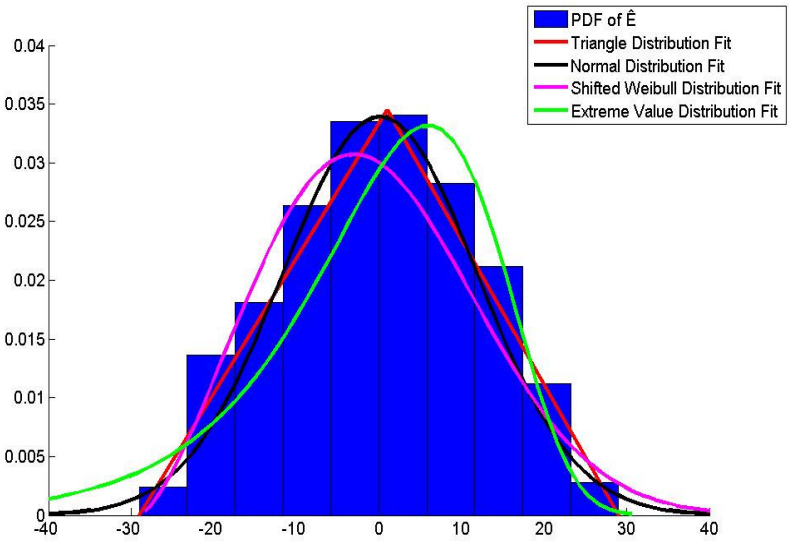

Figure 6.13: PDF of Estimated Delay and Distribution Fits (500 Hz Sinusoid on the Input)

6.3.5 Modelling the Delay Processes of the Devices

PDF of triangle distribution can be described as the convolution of two uniform distributions. Process of E is defined in equation 6.1 as $E = T_1 - T_2$. PDF of E can be derived from the PDFs of T_1 and T_2. PDF of E can be computed from the convolution integral shown in equation 6.5 [32].

$$f_E(k) = \int_{-\infty}^{\infty} f_{T_1}(k-x) f_{T_2}(-x) dx \qquad \textbf{(6.5)}$$

While $E = T_1 - T_2$, $f_{T_1} = f_{T_2}$ and f_{T_1} is a uniform distribution between 0 and T, convolution of f_{T_1} and f_{T_2} will be isosceles triangle as in Figure 6.14.

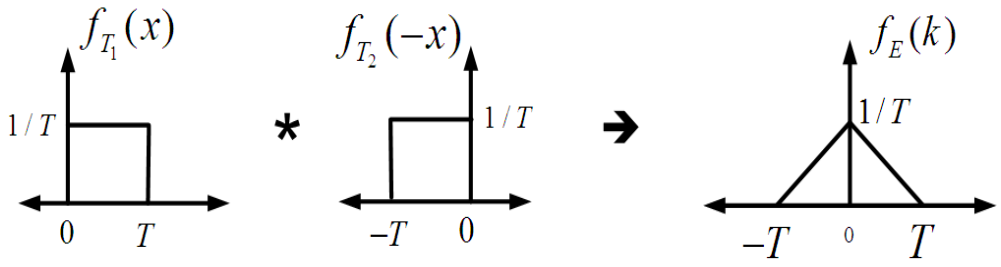

Figure 6.14: Convolution of f_{T_1} and f_{T_2}

Since Device 1 and Device 2 are identical in hardware and software, PDF of their delay processes f_{T_1} and f_{T_2} should be same. In Part 6.3.4 f_E is best fitted to the isosceles triangle distribution. As illustrated in Figure 6.14 f_E can be described the

convolution of two identical uniform distributions. During the tests absolute maximum value observed for $\hat{E}$ was 33μsec . Realization of 33μsec forces the bottom corners of isosceles triangle distribution to be at $\pm 33\mu$sec . In figure 6.15 corresponding f_{T_1} and f_{T_2} for the isosceles triangle distributed f_E can be seen.

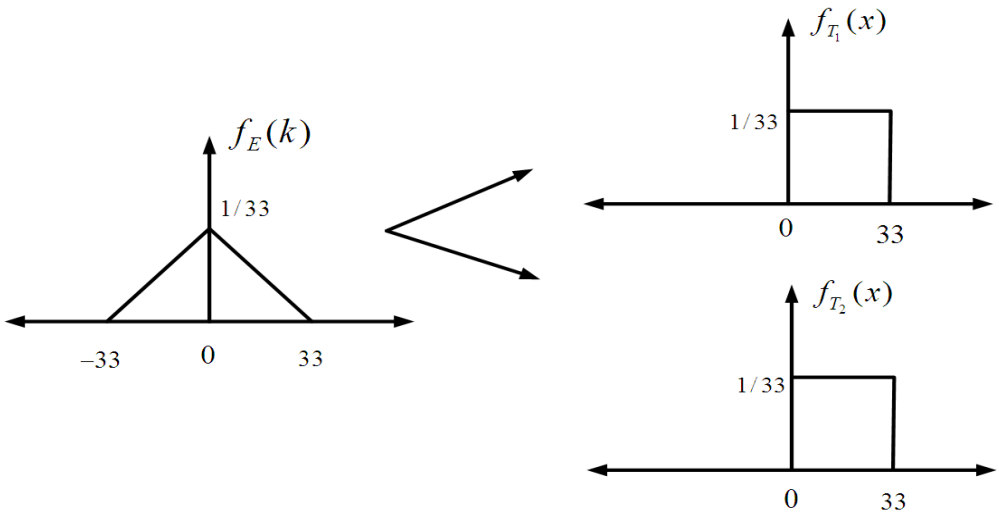

Figure 6.15: Corresponding f_{T_1} and f_{T_2} for the triangle distributed f_E

Consequently, it is showed that the delay processes of T_1 and T_2 introduced by the devices can be modeled as uniform distributed random processes between 0 and 33μsec .

6.3.6 MATLAB Graphical User Interface

A MATLAB graphical user interface is developed to display and save the collected data from the devices. This interface also calculates the delay between the acquired data and plots the distribution of delay. RS-232 communication port to get data from PAN Coordinator can be selectable from the pup up menu. In order to connect to the PAN Coordinator connect button is used. In Figure 6.16 graphical user interface can be seen. Interface displays the real-time data collected from the sensors in the first plot. It estimates the $\hat{E}$ and back shifts the Device 2's data in the order of $\hat{E}$ to overlap two device's data. It displays interpolated and back shifted data in the second plot. In this way the accuracy of the estimations can be checked visually. At the last plot it shows the histogram of $\hat{E}$.

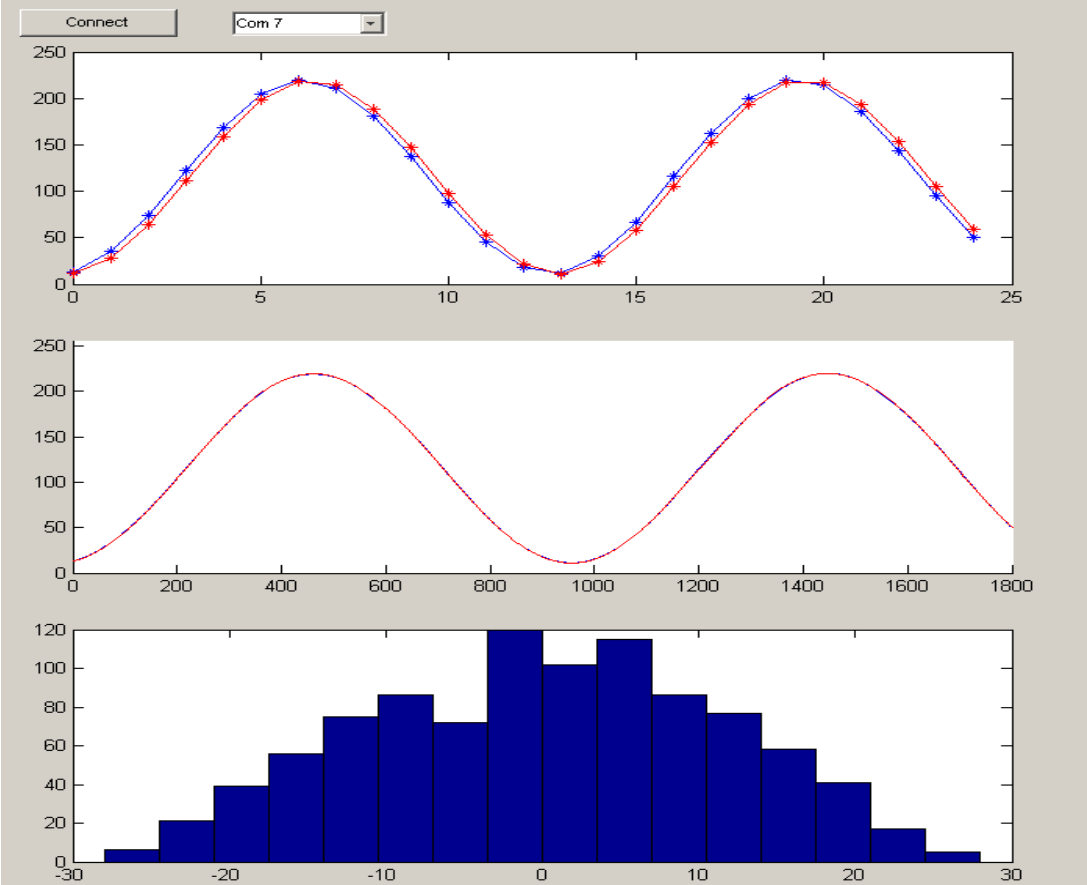

Figure 6.16: MATLAB Graphical User Interface to Display Sensor Data and Delay

6.4 Results

It is showed that sensor data from more than one sensor nodes in IEEE 802.15.4 networks can be time synchronized using the MLME-BEACON-NOTIFY.indication and a precise timer integrated in the node. Timer of the microcontroller unit is used to generate sampling intervals and reception of beacons started the sampling at each superframe. Because of running task scheduler in the devices response time to the beacon notify indication is random in behaviour. Uniform distribution is offered as a statistical model for this random behaviour.

7 CONCLUSION

Acquiring time synchronized sensor data from more than one sensor nodes has been successfully implemented in this study. On the way to this achievement, GTS performance of the 13192 EVK is measured and this performance is compared with the standard defined maximum performance in the MAC layer. With the results obtained from the measurements, using order two for superframe is decided. Superframe order two is a good compromise between bandwidth and latency.

Sensor nodes are synchronized with the beacons transmitted from the PAN coordinator. This synchronization is used to time stamp samples of the data from the sensors mounted on the devices. To generate constant sampling rate, timer of the microcontroller unit is used. At each beacon interval collected data from the sensors are sent using GTS. Measurements done about the performance of GTS, gave the idea of transmittable data in each beacon interval. Data acquired from the two sensor nodes are displayed in real-time with a MATLAB graphical user interface. Also delay introduced by the devices is inspected to reveal the error characteristics of synchronization.

REFERENCES:

[1] **I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci,** Wireless sensor networks: a survey, Computer Networks 38, 2002, pp. 393-422.

[2] **C. Intanagonwiwat, R. Govindan, D. Estrin,** Directed diffusion: a scalable and robust communication paradigm for sensor networks, proceedings of the ACM Mobicom'00, Boston, MA, 2000, pp. 56-67.

[3] **D. Estrin, L. Girod, G. Pottie, M. Srivastava,** "Instrumenting the World with Wireless Sensor Networks," IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001, vol. 4, no. 5, pp. 2033-2036, May 2001.

[4] **D. Estrin, R. Govindan, J. Heidemann, S. Kumar**, Next century challenges: scalable coordination in sensor networks, ACM MobiCom'99, Washington, USA, 1999, pp. 263-270.

[5] **A. Cerpa, J. Elson, M. Hamilton, J. Zhao,** Habitat monitoring: application driver for wireless communications technology, ACM SIGCOMM'2000, Costa Rica, April 2001.

[6] **P. Bonnet, J. Gehrke, P. Seshadri,** Querying the physical world, IEEE Personal Communications (October 2000) 10–15.

[7] **M. Ogawa et al.,** Fully automated biosignal acquisition in daily routine through 1 month, International Conference on IEEE-EMBS, Hong Kong, 1998, pp. 1947–1950.

[8] **B. Sibbald,** Use computerized systems to cut adverse drug events: report, CMAJ: Canadian Medical Association Journal 164 (13) (2001) 1878, 1/2p,1c.

[9] **E.M. Petriu, N.D. Georganas, D.C. Petriu, D. Makrakis,** V.Z. Groza, Sensor-based information appliances, IEEE Instrumentation and Measurement Magazine (December 2000) 31–35.

[10] **G.D. Abowd, J.P.G. Sterbenz,** Final report on the interagency workshop on research issues for smart environments, IEEE Personal Communications (October 2000) 36–40.

[11] **E. Shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, A. Chandrakasan,** Physical layer driven protocol and algorithm design for energy-

efficient wireless sensor networks, Proceedings of ACM MobiCom'01, Rome, Italy, July 2001, pp. 272–286.

[12] **A. Mainwaring *et al.*,** "Wireless Sensor Networks for Habitat Monitoring," *WSNA*, Atlanta, GA, Sept. 2002.

[13] **Z. Butler *et al.*,** "Networked Cows: Virtual Fences for Controlling Cows," *WAMES 2004*, Boston, MA, June 2004.

[14] "ARGO — Global Ocean Sensor Network," http://www.argo.ucsd.edu

[15] **R. Beckwith, D. Teibel, and P. Bowen.** "Pervasive Computing and Proactive Agriculture," *Adjunct Proc. PERVASIVE 2004*, Vienna, Austria, Apr. 2004.

[16] **F. Michahelles *et al.*,** "Applying Wearable Sensors to Avalanche Rescue," *Computers and Graphics*, vol. 27,no. 6, 2003, pp. 839–47.

[17] **G. Simon, A. Ledezczi, and M. Maroti.** "Sensor Network- Based Countersniper System," *Proc. SenSys*, Baltimore, MD, Nov. 2004.

[18] **Per H. Lehne**, A Brief Overview of Radio Technologies, Telektronikk, Vol 3/4.06, 2006

[19] **IEEE 802 LAN/MAN Standards Committee.** 18 November 2007 [online] – URL: http://www.ieee802.org/

[20] **IEEE 802.15 Working Group for WPAN**. 18 November 2007 [online] – URL: http://www.ieee802.org/15

[21] **Axel Sikora and Voicu F. Groza.** Coexistence of IEEE 802.15.4 with other Systems in the 2.4 GHz-ISM-Band. In Instrumentation and Measurement Technology Conference, IMTC 2005, Proceedings of the IEEE, 2005.

[22] **Anis Koubaa, Mario Alves, Eduardo Tovar,** "IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview", Version: 1.0 14 July 2005

[23] **"Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)",** IEEESA Standards Board

[24] **13192EVK Evaluation Kit User's Guide**, 802154EVKUG, Rev.1.2, 03/2007

[25] **Freescale 802.15.4 MAC/PHY Software User's Guide**, 802154MPSUG/D, Rev. 0.0, 11/2004

[26] **Freescale 802.15.4 MAC PHY Software Reference Manual**, 802154MPSRM, Rev. 1.4, 09/2006

[27] **David L. Mills,** Internet Time Synchronization: The Network Time Protocol, Global States and Time in Distributed Systems, IEEE Computer Society Press, 1994.

[28] **Lewis Girod and Deborah Estrin**, Robust range estimation using acoustic and multimodal sensing, In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001), March 2001.

[29] **H. Wang, L. Yip, D. Maniezzo, J.C. Chen, R.E. Hudson, J.Elson, and K.Yao,** A Wireless Time-Synchronized COTS Sensor Platform Part II-Applications to Beamforming, In Proceedings of lEEE CAS Workshop on Wireless Communications and Networking, Pasadena, CA, September 2002.

[30] **J. Elson, L. Girod, and D. Estrin**, Fine-grained network time synchronization using reference braodcasts , Technical Report UCLA-CS-020008, University of California, Los Angeles, May 2002.

[31] **MC13211/212/213/214 Reference Manual,** MC1321xRM, Rev. 1.1, 10/2006

[32] **A. Papoulis,** Probability, Random Variables, and Stochastic Processes –New York, 1991

APPENDIX 1: Maximum Payload and Goodput Values

Table A1.1: Maximum Payload and Goodput Values for 1 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	25	1	813.80	6.5104
2	85	1	1383.46	11.0677
3	85	2	1383.46	11.0677
4	118	3	1440.43	11.5234
5	102	7	1452.64	11.6210
6	112	13	1481.12	11.8489
7	118	25	1500.45	12.0035
8	118	50	1500.45	12.0035

Table A1.2: Maximum Payload and Goodput Values for 2 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	85	1	2766.92	22.1354
2	85	2	2766.92	22.1354
3	118	3	2880.85	23.0468
4	102	7	2905.27	23.2421
5	112	13	2962.23	23.6979
6	118	25	2999.79	23.9983
7	118	50	2999.79	23.9983
8	117	101	3005.21	24.0417

Table A1.3: Maximum Payload and Goodput Values for 3 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	118	1	3841.14	30.7291
2	85	3	4150.39	33.2031
3	109	5	4435.22	35.4817
4	109	10	4435.22	35.4817
5	116	19	4484.04	35.8728
6	116	38	4484.04	35.8728
7	118	75	4501.34	36.0107
8	118	150	4501.34	36.0107

Table A1.4: Maximum Payload and Goodput Values for 4 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	85	2	5533.85	44.2708
2	118	3	5761.71	46.0937
3	102	7	5810.54	46.4843
4	112	13	5924.47	47.3958
5	118	25	6001.79	48.0143
6	118	50	6001.79	48.0143
7	117	101	6010.43	48.0834
8	117	202	6010.43	48.0834

Table A1.5: Maximum Payload and Goodput Values for 5 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	115	2	7486.97	59.8958
2	115	4	7486.97	59.8958
3	115	8	7486.97	59.8958
4	115	16	7486.97	59.8958
5	115	32	7486.97	59.8958
6	117	63	7498.16	59.9853
7	118	125	7502.23	60.0179
8	118	250	7502.23	60.0179

Table A1.6: Maximum Payload and Goodput Values for 6 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	85	3	8300.78	66.4062
2	109	5	8870.44	70.9635
3	109	10	8870.44	70.9635
4	116	19	8968.09	71.7447
5	116	38	8968.09	71.7447
6	118	75	9002.68	72.0214
7	118	150	9002.68	72.0214
8	118	301	9032.69	72.2615

Table A1.7: Maximum Payload and Goodput Values for 7 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	105	3	10253.90	82.0312
2	105	6	10253.90	82.0312
3	117	11	10473.63	83.7890
4	117	22	10473.63	83.7890
5	117	44	10473.63	83.7890
6	116	89	10502.11	84.0169
7	118	175	10503.13	84.0250
8	118	351	10533.14	84.2651

APPENDIX 2: Measured Goodput Values

Table A2.1: Measured Goodput Values for 1 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	21	1	683.59	5.4687
2	81	1	1318.36	10.5468
3	82	2	1334.64	10.6770
4	82	4	1334.64	10.6770
5	99	7	1409.91	11.2792
6	99	14	1409.91	11.2792
7	99	28	1409.91	11.2792
8	101	55	1412.71	11.3016

Table A2.2: Measured Goodput Values for 2 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	82	1	2669.27	21.3541
2	82	2	2669.27	21.3541
3	82	4	2669.27	21.3541
4	99	7	2819.18	22.5585
5	99	14	2819.18	22.5585
6	94	28	2819.82	22.5585
7	101	55	2825.41	22.6033
8	102	109	2827.45	22.6196

Table A2.3: Measured Goodput Values for 3 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	52	2	3385.41	27.0833
2	82	3	4003.90	32.0312
3	102	5	4150.39	33.2031
4	102	10	4150.39	33.2031
5	99	21	4229.73	33.8378
6	102	41	4254.15	34.0332
7	102	82	4254.15	34.0332
8	102	164	4254.15	34.0332

Table A2.4: Measured Goodput Values for 4 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	82	2	5338.54	42.7083
2	82	4	5338.54	42.7083
3	99	7	5639.64	45.1171
4	99	14	5639.64	45.1171
5	99	28	5639.64	45.1171
6	101	55	5650.83	45.2067
7	102	109	5654.90	45.2392
8	102	219	5680.84	45.4467

Table A2.5: Measured Goodput Values for 5 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	102	2	6640.62	53.1250
2	82	5	6673.17	53.3854
3	95	9	6958.01	55.6640
4	102	17	7055.66	56.4453
5	102	34	7055.66	56.4453
6	101	69	7089.23	56.7138
7	102	137	7107.54	56.8603
8	102	274	7107.54	56.8603

Table A2.6: Measured Goodput Values for 6 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	82	3	8007.81	64.0625
2	102	5	8300.78	66.4065
3	102	10	8300.78	66.4065
4	99	21	8459.47	67.6757
5	102	41	8508.30	68.0664
6	102	82	8508.30	68.0664
7	102	164	8508.30	68.0664
8	102	329	8534.24	68.2739

Table A2.7: Measured Goodput Values for 7 GTS Allocation

SO=BO	Maximum payload size with fixed length packets (bytes)	Number of Packets in GTS	Goodput (bytes/s)	Goodput (kbits/s)
1	102	3	9960.93	79.6875
2	102	6	9960.93	79.6875
3	102	12	9960.93	79.6875
4	99	24	9960.93	79.6875
5	102	48	9960.93	79.6875
6	102	96	9960.93	79.6875
7	102	192	9960.93	79.6875
8	102	384	9960.93	79.6875

APPENDIX 3: Comparison of Maximum and Measured Goodputs

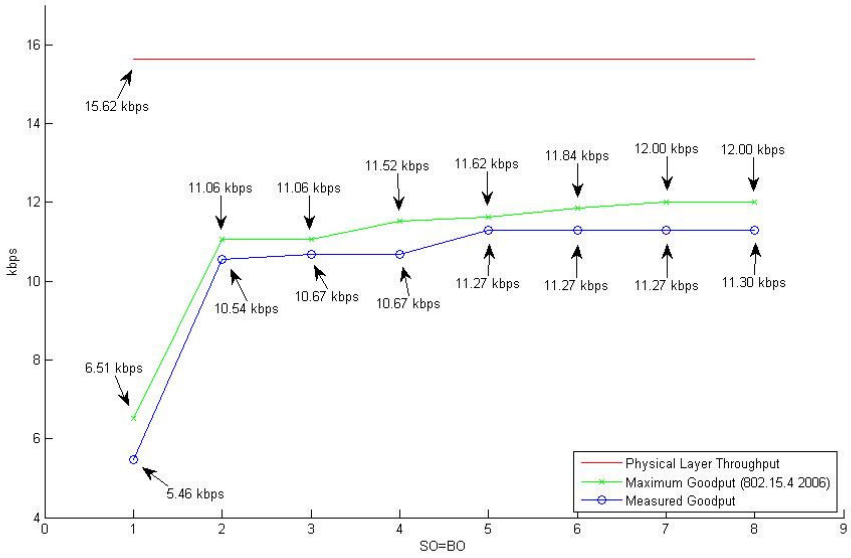

Figure A3.1: Comparison of Maximum and Measured Goodputs for 1 GTS Allocation

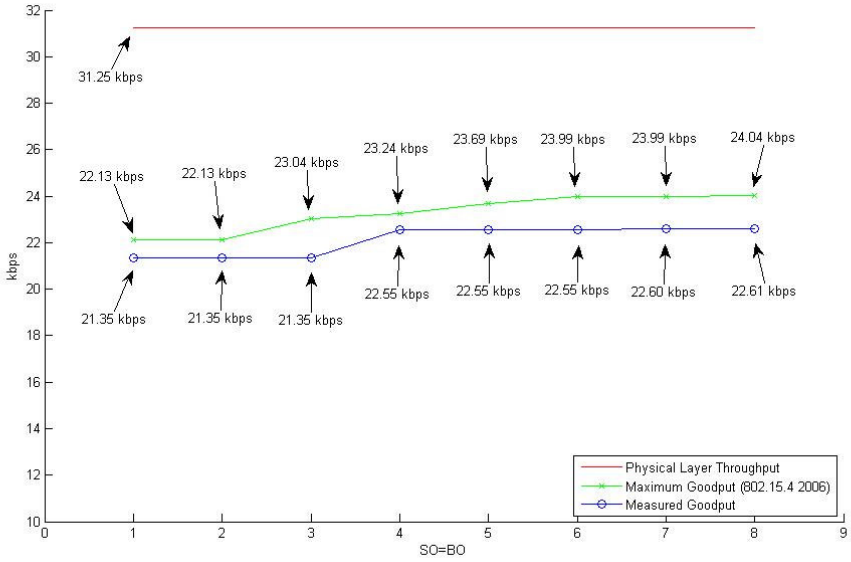

Figure A3.2: Comparison of Maximum and Measured Goodputs for 2 GTS Allocation

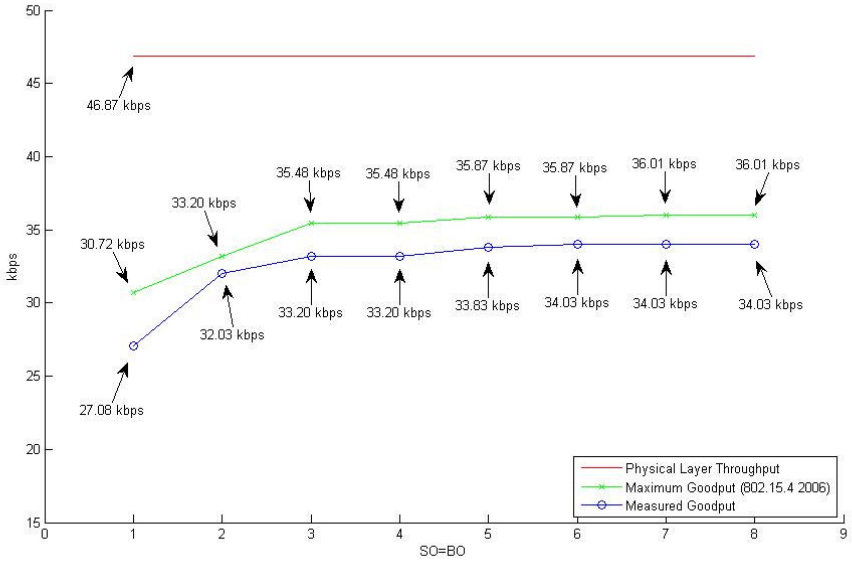

Figure A3.3: Comparison of Maximum and Measured Goodputs for 3 GTS Allocation

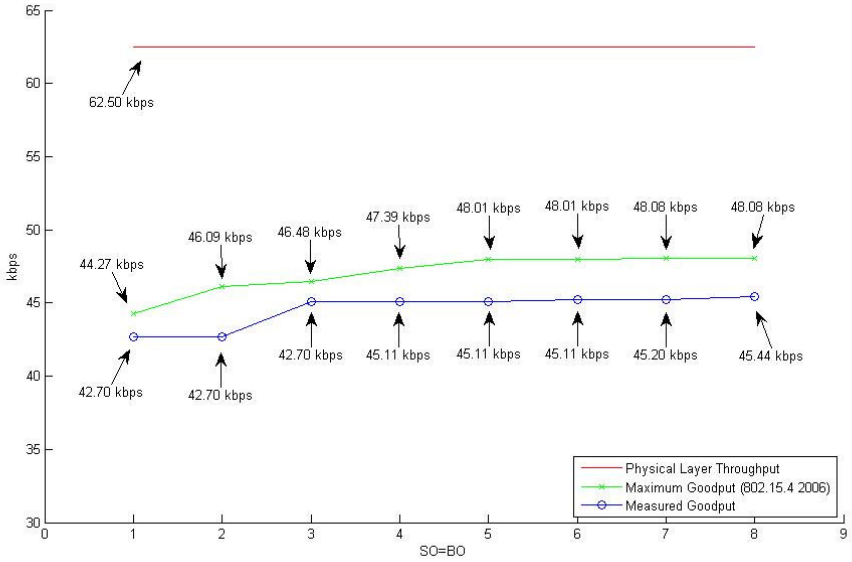

Figure A3.4: Comparison of Maximum and Measured Goodputs for 4 GTS Allocation

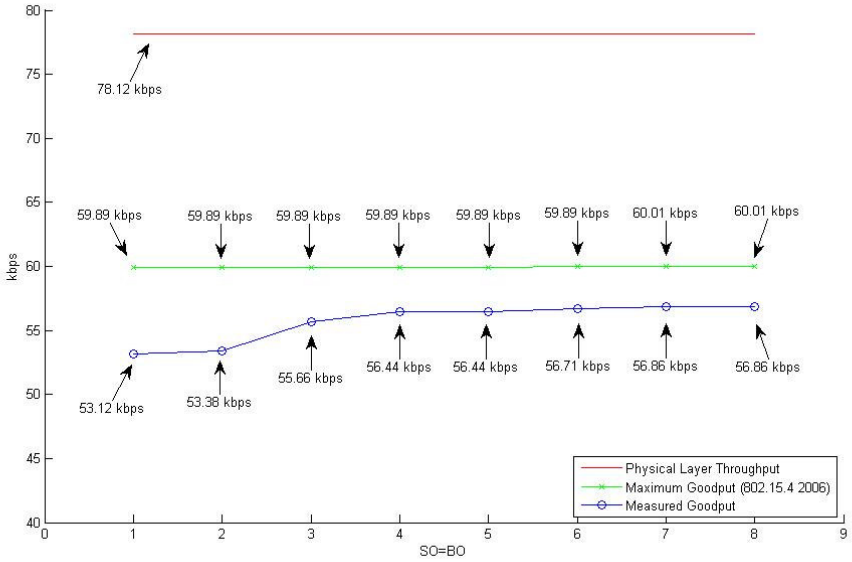

Figure A3.5: Comparison of Maximum and Measured Goodputs for 5 GTS Allocation

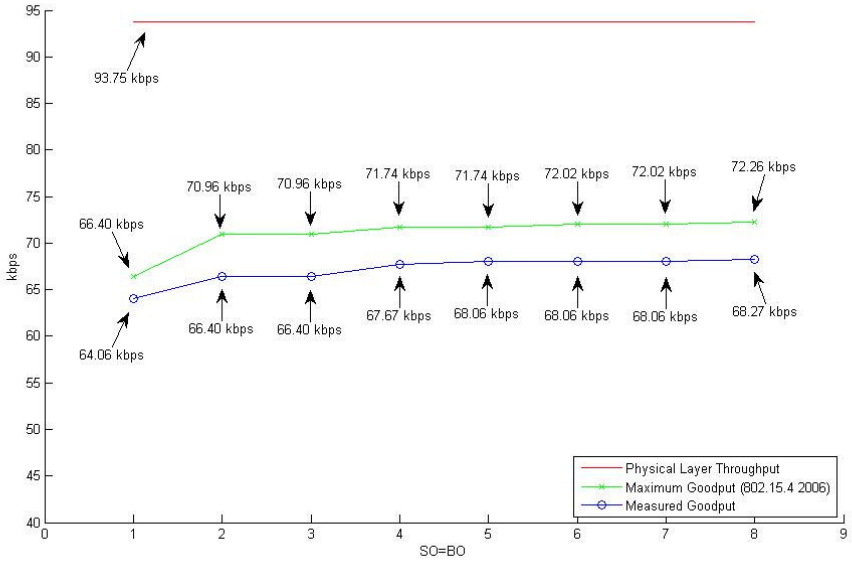

Figure A3.6: Comparison of Maximum and Measured Goodputs for 6 GTS Allocation

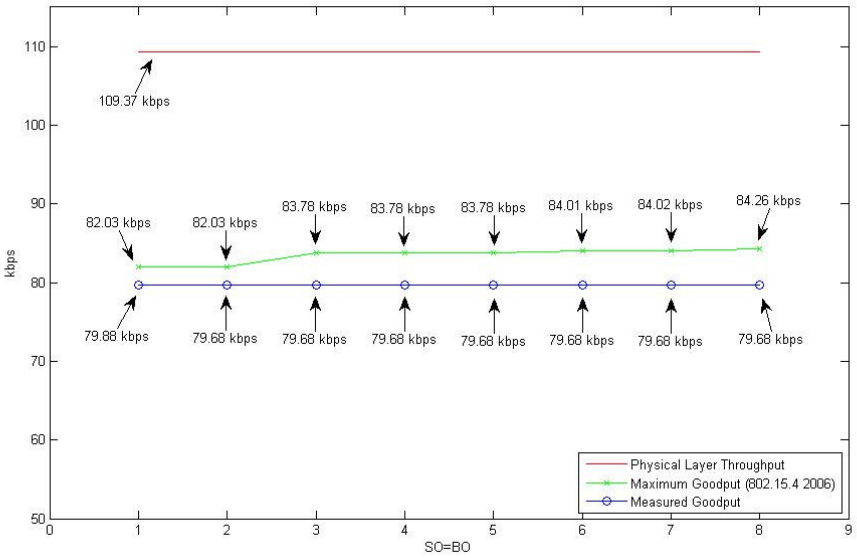

Figure A3.7: Comparison of Maximum and Measured Goodputs for 7 GTS Allocation

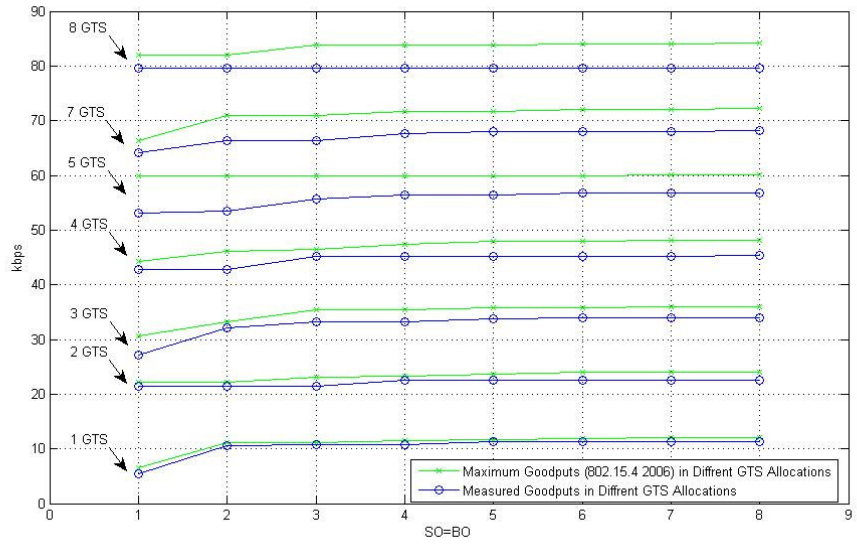

Figure A3.8: Comparison of Maximum and Measured Goodputs for All GTS Allocations

APPENDIX 4: Goodness of Fit to the Distributions of Delay

Table A4.1: Goodness of Fit (250 Hz Sinusoid on the Input)

Rank	Distribution	Goodness of Fit
1	Triangle	0.007
2	Normal	0.013
3	Shifted Weibull	0.024
4	Extreme Value	0.083

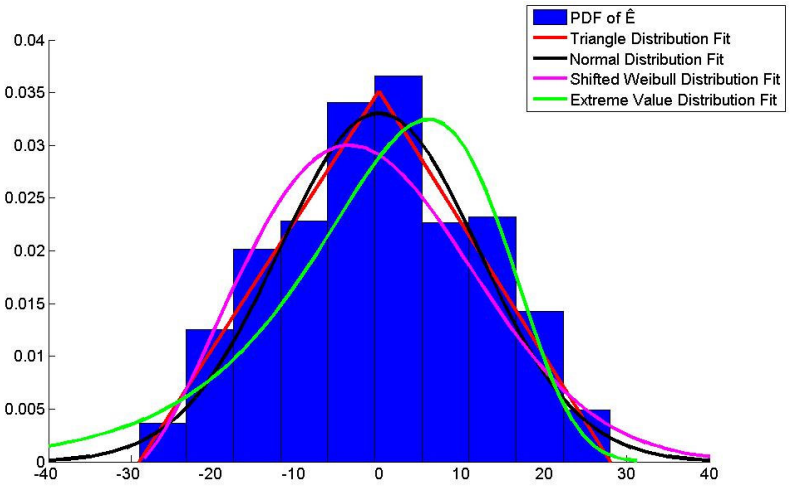

Figure A4.1: PDF of Estimated Delay and Distribution Fits (250 Hz Sinusoid on the Input)

Table A4.2: Goodness of Fit (500 Hz Sinusoid on the Input)

Rank	Distribution	Goodness of Fit
1	Triangle	0.004
2	Normal	0.013
3	Shifted Weibull	0.025
4	Extreme Value	0.075

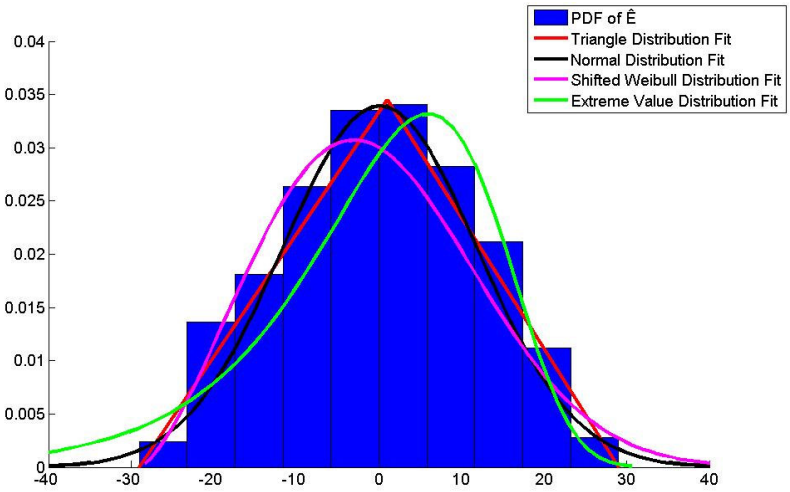

Figure A4.2: PDF of Estimated Delay and Distribution Fits (500 Hz Sinusoid on the Input)

Table A4.3: Goodness of Fit (1 kHz Sinusoid on the Input)

Rank	Distribution	Goodness of Fit
1	Triangle	0.012
2	Normal	0.014
3	Shifted Weibull	0.028
4	Extreme Value	0.082

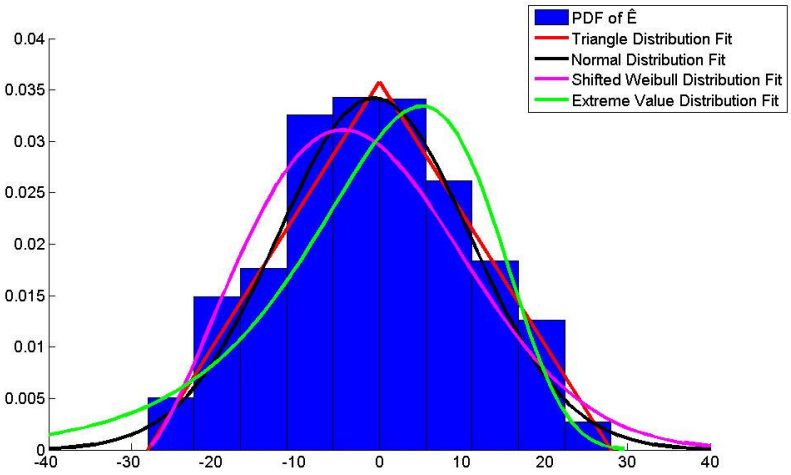

Figure A4.3: PDF of Estimated Delay and Distribution Fits (1 kHz Sinusoid on the Input)

Table A4.4: Goodness of Fit (2 kHz Sinusoid on the Input)

Rank	Distribution	Goodness of Fit
1	Triangle	0.005
2	Normal	0.011
3	Shifted Weibull	0.012
4	Extreme Value	0.102

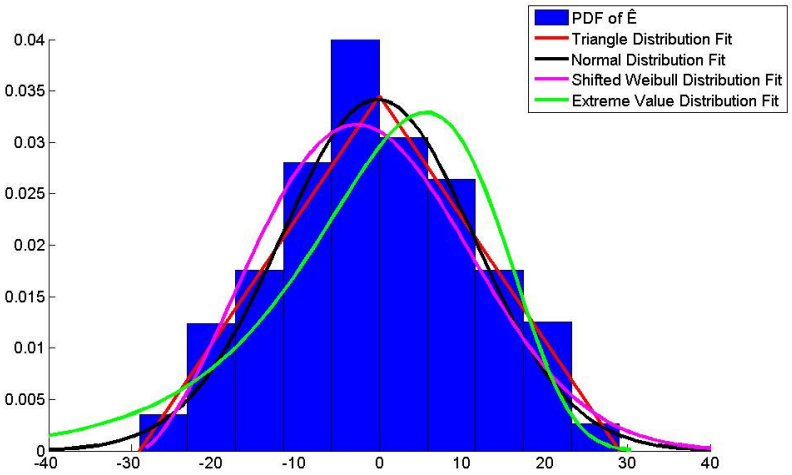

Figure A4.4: PDF of Estimated Delay and Distribution Fits (2 kHz Sinusoid on the Input)

APPENDIX 5: C Codes

Code Fragment for Accessing macBeaconTxTime:

```
/*Allocation of 24 bits long memory location to store macBeaconTxTime attribute*/
uint8_t beaconTxTime[3];
/*Allocation of data structure which will be passed to the MLME SAP handler*/
mlmeMessage_t mlmeGet;
/* Storing the Desired Message Type in the Data Structure*/
mlmeGet.msgType = gMlmeGetReq_c;
/* Storing the Desired PIB Attribute in the Data Structure*/
mlmeGet.msgData.getReq.pibAttribute = gMacPibBeaconTxTime_c;
/* Storing the Address of Allocated Memory Location in the Data Structure*/
mlmeGet.msgData.getReq.pibAttributeValue = beaconTxTime;
/* Sending the MLME-GET.request to the MLME*/
MSG_Send(NWK_MLME, &mlmeGet);
```

The use of MLME-SET.request is only differs in the value of msgType and the pibAttributeValue parameter in the message. In set request, pibAttributeValue is a return value.

Code Fragment for GTS.request:

```
/*Allocation of data structures which will be passed to the MLME SAP handler*/
mlmeMessage_t *pMsg;
mlmeGtsReq_t *pGtsReq;
/* Allocate a message for the MLME */
pMsg = MSG_AllocType(mlmeMessage_t);
/*Check for allocation*/
if(pMsg != NULL)
{
        /* This is a MLME-GTS.req command */
        pMsg->msgType = gMlmeGtsReq_c;
        /* Create the GTS request message data. */
        pGtsReq = &pMsg->msgData.gtsReq;
        pGtsReq->securityEnable = FALSE;
        /*GTS allocation, length 1, transmit only*/
        pGtsReq->gtsCharacteristics = 0x21;
}
/* Sending the MLME-GTS.request to the MLME*/
MSG_Send(NWK_MLME, pMsg);
```

Code Fragment to Send Data Using GTS:

```
uint8_t payloadLength = 10;
/*Data that will be sent*/
uint8_t *payload="1234567890";
/*Allocate a message to send MCPS*/
nwkToMcpsMessage_t* pPacket = MSG_AllocType(nwkToMcpsMessage_t);
/*Check for allocation*/
if(pPacket != NULL)
{
        pPacket->msgType = gMcpsDataReq_c;
        memcpy(pPacket->msgData.dataReq.msdu,payload,payloadLength);
        memcpy(pPacket->msgData.dataReq.dstAddr, coordInfo.coordAddress, 8);
        memcpy(pPacket->msgData.dataReq.srcAddr, myAddress, 8);
         memcpy(pPacket->msgData.dataReq.dstPanId, coordInfo.coordPanId, 2);
        memcpy(pPacket->msgData.dataReq.srcPanId, coordInfo.coordPanId, 2);
        pPacket->msgData.dataReq.dstAddrMode = coordInfo.coordAddrMode;
        pPacket->msgData.dataReq.srcAddrMode = myAddrMode;
        pPacket->msgData.dataReq.msduLength = payloadLength;
        /* Request GTS for the data packet */
        pPacket->msgData.dataReq.txOptions = gTxOptsGts_c;
        /* Give the data packet a handle. The handle is
        returned in the MCPS-Data Confirm message. */
        pPacket->msgData.dataReq.msduHandle = msduHandle++;
        /* Send the Data Request to the MCPS */
        NR MSG_Send(NWK_MCPS, pPacket);
        /* Prepare for another data buffer */
        pPacket = NULL;
}
```

Code Fragment to Set macAutoRequest Value to False:

```
/*Message Allocation*/
mlmeMessage_t *pMsgOut = MSG_AllocType(mlmeMessage_t);
uint8_t value = FALSE;
/*This is a MLME-SET.Request*/
pMsgOut->msgType = gMlmeSetReq_c;
pMsgOut->msgData.setReq.pibAttribute = gMacPibAutoRequest_c;
pMsgOut->msgData.setReq.pibAttributeValue = &value;
/*Send MLME-SET.Request to the MLME*/
NR MSG_Send(NWK_MLME, pMsgOut);
```

Code Fragment to Get Beacon Sequence Number (BSN):

```
/*Allocation of memory location*/
uint8_t BSN;
/* Get BSN Value*/
BSN=nwkMessage_t *)pMsgIn)->msgData.beaconNotifyInd.bsn;
/*After each beacon notification indication message pBufferRoot should be freed*/
MSG_Free(((nwkMessage_t*)pMsgIn)->msgData.beaconNotifyInd.pBufferRoot);
```

Code Fragment to Start the Timer:

```
/*Timer Start*/
TPM2CNTL = 0x00;
TPM2CNTH = 0x00;
/* 200 us interrupt interval*/
TPM2MODH = 0x00;
TPM2MODL = 0xC8;          /*0xC8=200 */
/* Timer overflow interrupt is active, prescaller is 16(1us), clock source is BUSCLK*/
TPM2SC   = 0x4C;
```

Code Fragment to Initialize the Analog Digital Converter:

```
ATDPE=0x83; /* enable desired ADC channels (AD0, AD1, AD7 on)*/

ATDC=0xF7; /*set prescale to 7, 8 bit resolution, unsigned, right justified*/
```

Code Fragment to Start Analog Digital Conversion:

```
ATDSC = 0x01; /*read X channel*/
```

Code Fragment to Get Results from Analog Digital Converter:

```
if((ATDSC & 0x80) == 0x80)

{

adc_result = ATDRH;

}
```

RESUME

Cengiz Gezer was born in Istanbul in 1981. He was graduated from Eskişehir Kılıçoğlu Anatolian High School in 1999. He received the B.Sc. degree in Electrical - Electronics Engineering from Eskişehir Osmangazi University in 2005.