國立交通大學

資訊科學與工程研究所

碩 士 論 文

在 NCTUns 平台上模擬 WiMAX 點對多點網路

Simulating WiMAX PMP Networks over the NCTUns Network Simulator

研 究 生：林仕盈

指導教授：王協源　教授

中 華 民 國　九 十 五　年 六 月

摘要

IEEE 802.16 (WiMAX) 是目前最重要的新興無線技術之一，係由 Intel、Nokia 等國際大廠所共同制定推動的通訊標準。它被預期將成為下一代網路無線的主流，提供家庭、公司、及各種通訊所需的服務。IEEE 802.16 規格定義了 PMP 與 Mesh 兩種運作模式。PMP 模式主要的運用在"最後一哩"的傳輸，用來取代傳統的固接式網路，如 xDSL 與 T1；Mesh 模式則屬於網路的基礎建設，提供高速可靠的骨幹網路。

IEEE 802.16-2001 是最初訂定的 802.16 標準，到現在發展的時間已經超過六年了。然而卻沒有一個公開的網路模擬器能提供 802.16 網路的模擬。我們的目標在 NCTUns 這個模擬平台上開發屬於 802.16d 的模組。根據我們的成果，希望可以幫助研究者更方便快速地完成他們的研究。

Abstract

The IEEE 802.16 (WiMAX) standard, supported by several major companies such as Intel and Nokia, is one of the most important standards for broadband wireless access technologies in the future. It is expected to become the mainstream of network technology for home user, business, and telecommunication in the future. The standard defines two operation modes – PMP (Point-to-MultiPoint) and Mesh. The PMP mode aims to replace traditional wired last-mile access network, such as xDSL and T1, with a novel wireless access technology. The Mesh mode was designed for construct infrastructure network supporting high-speed reliable backbone networks.

The first IEEE 802.16-2001 standard was developed from 2001. The IEEE 802.16 standard family has been developed over six years. However, so far there is not a generic network simulator for simulating such networks. Our work is based on the NCTUns network simulator to develop a protocol suite of WiMAX network simulator in accordance with the IEEE 802.16d Standard. Based on our work, researchers can easily analyze what they research and save more experiment time.

致謝

首先相當感謝指導老師王協源教授這兩年多以來的指導，讓我接觸到許多先進的技術及知識，特別是瞭解到開發一個系統的辛苦以及其中的困難。

感謝各位口試委員撥冗來到交通大學進行口試，您們的意見及建議，使得這篇論文能夠更佳完整。

感謝周智良學長及林志哲學長，在這過程中，不論是專業的知識，或是生活上心態的調適，你們都提供我相當大的意見及幫助。

感謝我的 partner 朱漢威，一直以來，和你的合作都是非常的歡樂，這是相當寶貴的經驗。最後也勉勵實驗室的學弟們繼續努力，期望你們也能夠順利地完成學業。

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

As wireless networks become popular, many different technologies have been proposed to enhance the quality of services provided by such wireless networks. Nowadays, WiMAX (Worldwide Interoperability for Microwave Access) is one of the most promising technologies. It has been standardized by the IEEE 802.16 working group.

The first IEEE 802.16-2001 standard was developed in 2001 using frequencies between 10 GHz and 66 GHz for line of sight (LOS) transmissions. Its amendment IEEE 802.16a (802.16-2003) was defined later in 2003 using lower frequencies from 2 to 11 GHz for non-line of sight (NLOS) transmissions. The integration of above two standards was published in 2004 as IEEE 802.16-2004. The IEEE 802.16-2004 standard defines the air interface for fixed broadband wireless access (FBWA) systems for local and metropolitan networks (LAN and MAN). In the IEEE 802.16 Standard, the specifications of medium access control (MAC) layer and physical (PHY) layer are defined. The MAC layer in IEEE 802.16 has two modes, point to multi-point (PMP) mode and mesh mode. In terms of the physical layer, four different technologies, SC, SCa, OFDM, and OFDMA are defined in this standard. In this thesis, we stress on the PMP mode with OFDM physical layer specification.

Currently, network infrastructures in most countries are wired systems like xDSL and cable modem systems. Wired systems have high costs on the deployment and maintenance of cables. Since WiMAX uses broadband radio technology, it can be easily and conveniently deployed with a lower cost than traditional wired systems.

The IEEE 802.16 standard family has been developed over six years. However, so far there is no generic network simulator for simulating such networks. Some

researchers have developed their own simulators for WiMAX networks. But their simulators are not open to the public. Our work is based on the NCTUns network simulator to develop a protocol suite of WiMAX network simulator in accordance with the IEEE 802.16d Standard.

The NCTUns network simulator has the following features and advantages. First, real-world network protocols can be used to generate accurate simulation results. Second, real-world application programs can be run on a simulated network to generated realistic traffic. Thirdly, it provides a very easy-to-use and fully-integrated GUI environment. Finally, NCTUns uses open source architecture to allow a researcher to easily add his/her own protocol module into the simulator. Based on those features, we developed protocol modules simulating the IEEE 802.16d standard on NCTUns. These protocol modules will be released without charge and should be helpful for researchers who are interested in this technology.

The rest of this thesis is organized as follows. In chapter 2, we describe the background of IEEE 802.16 protocol standards and the NCTUns network simulator. In chapter 3, we briefly introduce the IEEE 802.16 Standard, which is divided into two parts - MAC and PHY layers. In chapter 4 and 5, we describe the design and implementation of our work. In chapter 6, we evaluate the simulation results of our protocol modules. Finally, in chapter 7 we discuss the future work and conclude this thesis.

Chapter 2. Background

IEEE 802.16 is a next-generation broadband wireless access system for metropolitan area networks. It is expected to become the mainstream of network technology for home user, business, and telecommunication in the future. The IEEE 802.16 Working Group is responsible for the development and amendments of IEEE Standard 802.16. Members in this working group were involved in 802.16 Standard before 2000, and had developed the following important standards 802.16 (2001), 802.16c (2002), 802.16a (2003), 802.16d (2004), and 802.16e (2005). There are some new proposed specifications related to WiMAX which are from 802.16f to 802.16k. Since the IEEE Standard 802.16e was just released, we focus on the IEEE Standard 802.16d in the thesis.

The IEEE Standard 802.16 is the first standardized version of IEEE 802.16. It operates in the frequency ranges of 10-66 GHz, and its coverage can be from 1 to 3 miles (5 km). It utilizes 20 to 28 MHz for a channel and provides a data rate up to 134 Mbps. Since it works on a high frequency band, LOS transmission is required between base station and subscriber station. Therefore, it is not suitable for home or indoor users.

The IEEE 802.16a Standard is the amendment of 802.16, which operates in the frequency band below 11 GHz to support NLOS transmission. It uses OFDM (Orthogonal Frequency Division Multiplexing) to enlarge the transmission coverage up to 4~6 miles (10 km) and enhance the resistance for multi-path effect. Its data rates can reach up to 70 Mbps at 20 MHz bandwidth.

The IEEE 802.16-2004 standard specifies the air interface, including the MAC layer and multiple PHY layer specifications, of fixed BWA systems supporting

multiple services. It consolidates IEEE Standard 802.16, IEEE Standard 802.16a, and IEEE Standard 802.16c. It retains all modes and major features without adding new modes.

Figure 2-1 illustrates the reference model of IEEE Standard 802.16 and the scope of this standard. The standard defines the data/control plane of MAC and PHY layers. Vendors can set up the management plane in their own ways. The MAC layer comprises three sub-layers, Service-Specific Convergence Sub-layer (CS or SSCS), MAC Common Part Sub-layer (CPS), and Security Sub-layer. The SAP (service access point) is provided by lower layer in protocol stack to the next higher layer for a way to access next protocol layer.

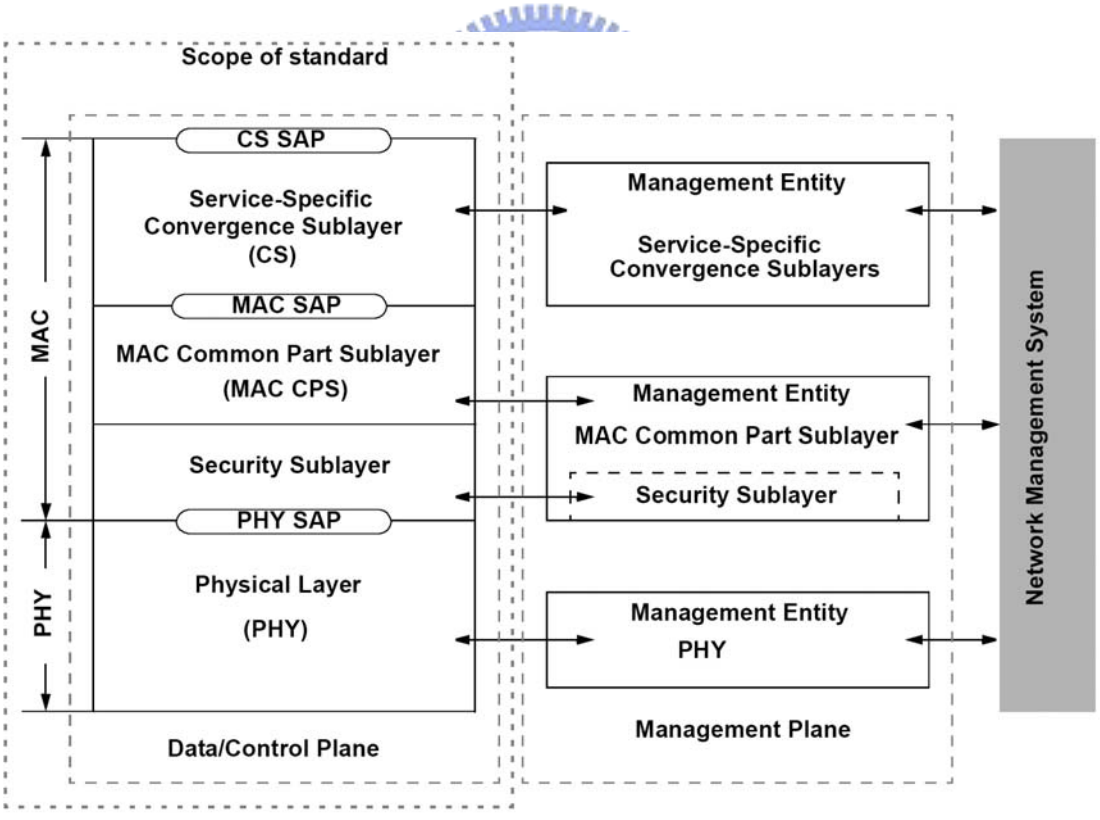

Figure 2-1 Reference model of IEEE Std 802.16

2.1. Medium Access Control Layer

2.1.1. Service Specific CS

The service specific CS resides on top of the MAC CPS and performs the following functions:

(1) accepting protocol data units (PDUs) from the higher layer

(2) performing classification of higher-layer PDUs into appropriate connection

(3) processing (if required) the higher-layer PDUs based on the classification

(4) delivering CS PDUs to the appropriate MAC SAP

(5) receiving CS PDUs from the peer entity

In the standard, two CS specifications are provided: the asynchronous transfer mode (ATM) CS and the packet CS. The ATM CS accommodate ATM network with transformation of the 802.16 MAC layer. In the packet CS, four different subdivisions are used for interfacing with various protocols. The CS converts different upper-layer protocol data unit (PDU) into the format regulated by the MAC CPS. Therefore, the MAC CPS does not need to understand the format of various data payload. Figure 2-2 shows the supported CS types in IEEE 802.16-2004. The particular CS can recognize corresponding upper layer protocol data unit. For example, the IPv4/IPv6 CS classify PDUs into connections according to the IP header.

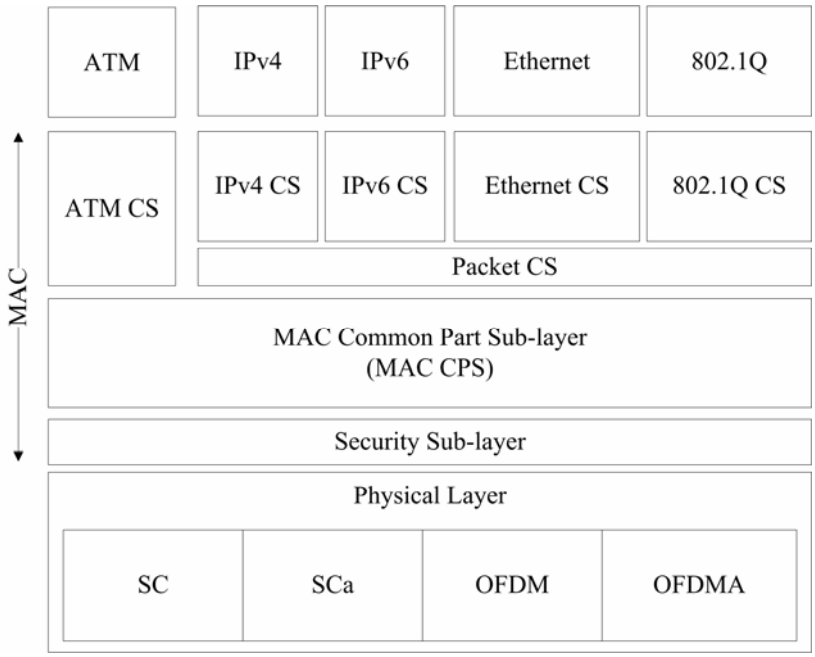

Figure 2-2 Various CSs and PHYs of 802.16

2.1.2. MAC Common Part Sub-layer

The MAC CPS provides the core MAC functionality of system access, including bandwidth allocation, connection establishment, and connection maintenance. It receives PDUs from various CSs, through the MAC SAP, then classifies to particular MAC connection. The MAC CPS encapsulates information into its own MAC PDU format. Figure 2-3 and Figure 2-4 show the format of MAC PDU and generic header. The MAC PDU starts with a 6-bytes generic MAC header, followed by an optional payload. The variable-length payload field carries either upper-layer protocol data units or MAC layer management messages. After payload, a MAC PDU may contain a CRC to protect overall PDU, including MAC header and payload.

The MAC generic header contains the length, connection, and sub-header information of the MAC PDU. The type field indicates the sub-headers and special payload types present in the message payload. Six kinds of sub-headers and payload

types are defined in the standard: mesh sub-header, ARQ feedback payload, extended type, fragmentation sub-header, packing sub-header, and FAST-FEEDBACK allocation sub-header/grant management sub-header. Mesh sub-header indicates the use of Mesh-mode connection. Fragmentation sub-header and packing sub-header help to utilize transmission allocation efficiently Fragmentation sub-header can help to divide one message into multiple MAC PDUs. Packing sub-header can help to concatenate multiple messages into one MAC PDU. The FAST-FEEDBACK allocation and grant sub-header are used in downlink and uplink direction respectively for bandwidth request and allocation. They can share the same field in the generic header to indicate their presence. ARQ feedback indicates that the payload enables automatic repeat request capability. The ARQ mechanism is a part of the MAC, which is optional for implementation. Extended type is used for extending the capability of fragmentation and packing. The CI (CRC Indication) indicates the presence of CRC at the end of MAC PDU.

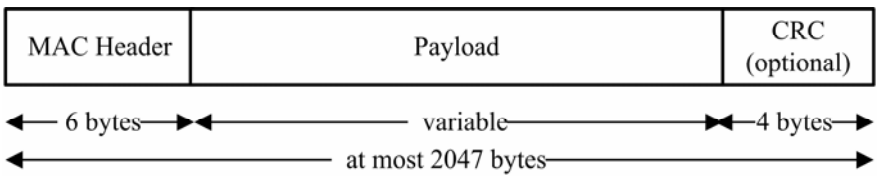

Figure 2-3 MAC PDU formats

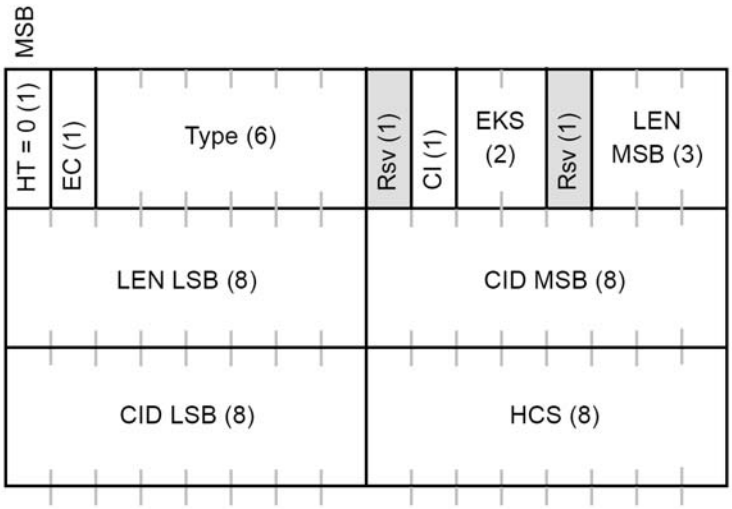

Figure 2-4 Generic MAC Header

All transmissions in the 802.16 PMP-mode are connection oriented. Connections are distinguished by different connection identification (CID). Each connection is responsible to different types of messages. Some connections use well-known CID values so that they can be shared among all SS in the WiMAX network, such as initial ranging and broadcast connection. Some connections are assigned to subscriber stations (SS) by the base station (BS). At SS initialization, the basic and the primary management connection shall be established between the SS and the BS and the secondary management connection may be optionally generated. These management connections are use for transmitting management messages defined in IEEE 802.16 standard. Transport CID is assigned to a connection used to deliver upper-layer protocol data unit, such as IP packets and Ethernet frames. The CID usages defined in IEEE 802.16d are illustrated in Table 2-1.

Table 2-1 CIDs

CID	**Value**	**Description**
Initial Ranging	0x0000	Used by SS and BS during initial ranging process.
Basic CID	0x0001– m	The same value is assigned to both the DL and UL connection.
Primary Management CID	m+1 – 2m	The same value is assigned to both the DL and UL connection.
Transport CIDs and Secondary Management CIDs	2m+1– 0xFEFE	For the secondary management connection, the same value is assigned to both the DL and UL connection.
AAS initial ranging CID	0xFEFF	A BS supporting AAS shall use this CID when allocating a Initial Ranging period for AAS devices.
Multicast polling CIDs	0xFF00– 0xFFFD	An SS may be included in one or more multicast polling groups for the purposes of obtaining bandwidth via polling. These connections have no associated service flow.
Padding CID	0xFFFE	Used for transmission of padding information by SS and BS.
Broadcast CID	0xFFFF	Used for broadcast information that is transmitted on a downlink to all SS.

The MAC management message exchanging information between base station and subscriber station shall be carried in the payload of the MAC PDU. All MAC management message begins with a Management Message Type field and may contain additional fields. Optional information of the management message can be carried in the TLV encoding. The type field of the TLV encoding depends on the type of the management message. The MAC management messages can deliver network and scheduling information or negotiate system status between base station and subscriber station. The format of the management message and the TLV encoding are given in Figure 2-5 and Figure 2-6.

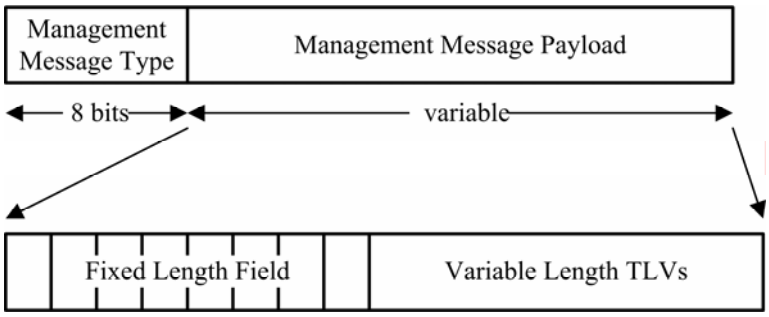

Figure 2-5 MAC Management Message format

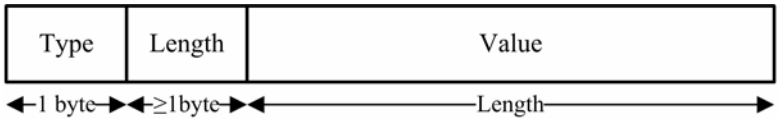

Figure 2-6 TLV encoding

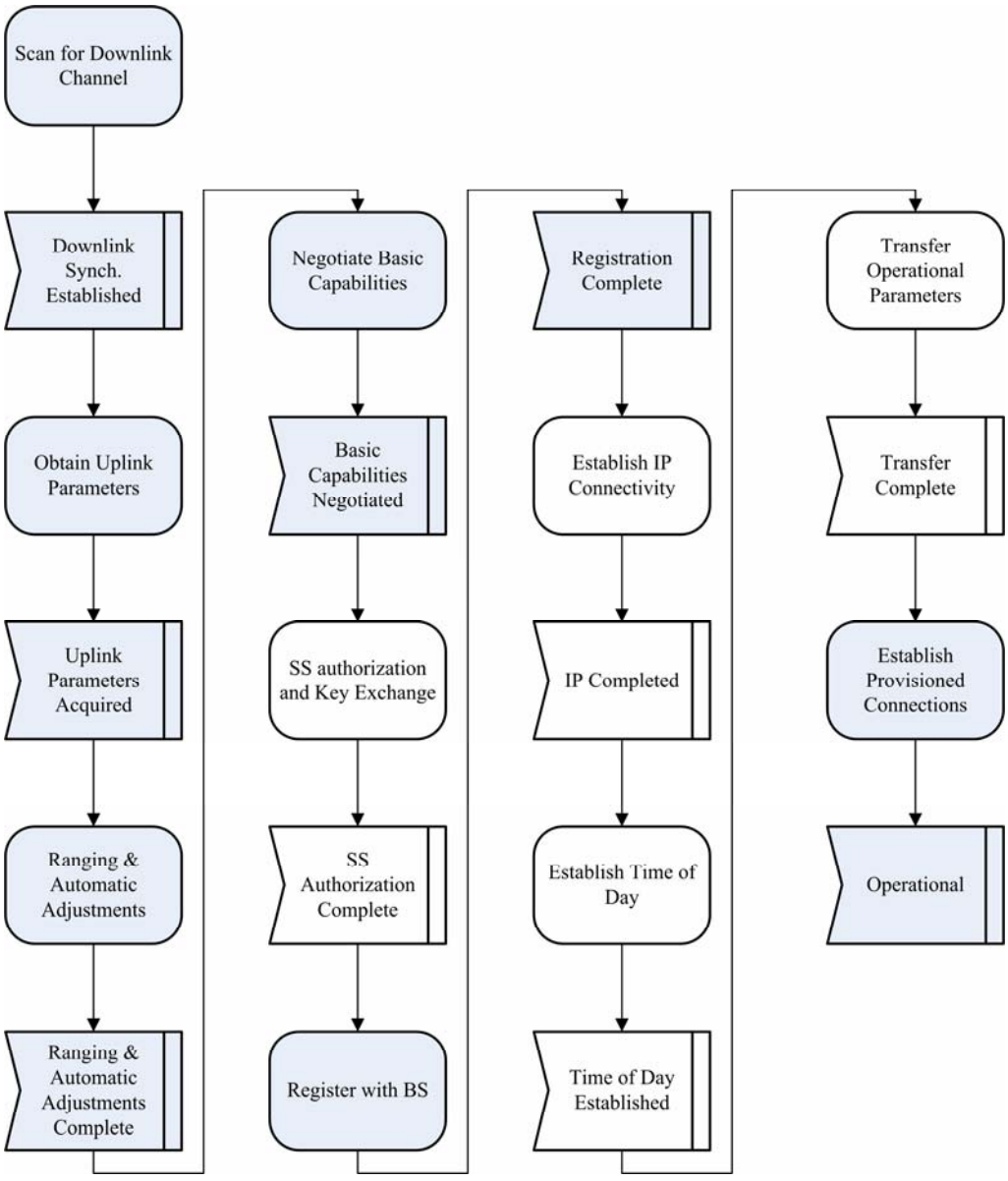

Figure 2-7 SS initialize overview

Table 2-2 Common Management Messages for PMP mode

Type	Message name	Message description	Connection	By
0	UCD	Uplink Channel Descriptor	Broadcast	BS
1	DCD	Downlink Channel Descriptor	Broadcast	BS
2	DL-MAP	Downlink Access Definition	Broadcast	BS
3	UL-MAP	Uplink Access Definition	Broadcast	BS
4	RNG-REQ	Ranging Request	Initial Ranging or Basic	SS
5	RNG-RSP	Ranging Response	Initial Ranging or Basic	BS
6	REG-REQ	Registration Request	Primary Management	SS
7	REG-RSP	Registration Response	Primary Management	BS
11	DSA-REQ	Dynamic Service Addition Request	Primary Management	BS/SS
12	DSA-RSP	Dynamic Service Addition Response	Primary Management	BS/SS
13	DSA-ACK	Dynamic Service Addition Acknowledge	Primary Management	BS/SS
26	SBC-REQ	SS Basic Capability Request	Basic	SS
27	SBC-RSP	SS Basic Capability Response	Basic	BS

Every time a subscriber station boots or attaches to the WiMAX network, the network entry procedures was activated. The applicable network entry procedure is provided for entering and registering a new SS or a new node to the network. It also supports establishing basic connections between base station and subscriber stations for operational purpose. Figure 2-7 shows the network entry procedure overview for SS initialization, and Table 2-2 gives common management messages used in PMP mode for network entry and operation. As shown in Figure 2-7, the procedure contains the following phases:

(a) Scan for downlink channel and establish synchronization with BS

(b) Obtain transmit parameters from UCD message

(c) Perform ranging

(d) Negotiate basic capabilities

(e) Authorize SS and perform key exchange

(f) Perform registration

(g) Establish IP connectivity

(h) Establish time of day

(i) Transfer operational parameters

(j) Set up connections

Since phases (g), (h), and (i) only performs if the SS is a managed SS, these phases are operational and do not implement simulation.

First, the SS scans for downlink channel and gathers channel information from DCD and DL-MAP messages. After downlink channel synchronization, the SS waits for a UCD message from the BS in order to retrieve a set of transmission parameters for a possible uplink channel. The SS then wait for UL-MAP message to acquire the information of uplink allocations for ranging. The SS sends RNG-REQ message to the BS till ranging is success. In this phase, the BS also replies RNG-RSP message to indicate the ranging progress. In the meantime, the BS also assigns basic and primary management connection identifications to this new SS. After ranging, the SS and BS use SBC-REQ and SBC-RSP messages via basic connection to exchange capabilities and to negotiate transmission parameters. We skip the authorization and key exchange phase that belongs to the security sub-layer. The registration procedure is then performed for that the SS is allowed entering into the network. The REG-REQ and REG-RSP messages are used in registration phase. In the connection set-up phase, one or many data transport connection will be established according to configuration of the BS or contract between user and ISP. Finally, the network entry procedure is done and the SS enters operational state.

2.1.3. Security Sub-layer

In the 802.16 standard, the MAC layer contains security sub-layer to provide the functionality of authentication, secure key exchange, and encryption. Since it was an optional feature and it does not affect performance result or operations of network communication, we do not focus on the security sub-layer and bypass it in our simulations.

2.2. Physical Layer

The IEEE 802.16 standard uses frequencies from 10 to 66 GHz for LOS transmission, and frequencies from 2 to 11 GHz for NLOS transmission. In the standard, messages can be modulated using different modes, including single carrier, OFDM (orthogonal frequency division multiplexing), and OFDMA (orthogonal frequency division multiple access). Table 2-3 shows the supported physical specifications defined in the IEEE 802.16 standard. In this thesis, we focus on the WirelessMAN-OFDM with TDD (time division duplex) mode for simulating the most commonly used environment of WiMAX network. The WirelessMAN-HUMAN works same as other three NLOS specifications for unlicensed spectrum. There are only four kinds of physical technologies defined in the standard.

Table 2-3 Variants of WiMAX PHY (Source: WiMAX Handbook)

Designation	**Function**	**LOS/ NLOS**	**Frequency**	**Duplexing Alternative(s)**
WirelessMAN-SC	Point-to-point	LOS	10-66 GHz	TDD, FDD
WirelessMAN-SCa	Point-to-point	NLOS	2-11 GHz	TDD, FDD
WirelessMAN-OFDM	Point-to-multipoint	NLOS	2-11 GHz	TDD, FDD
WirelessMAN-OFDMA	Point-to-multipoint	NLOS	2-11 GHz	TDD, FDD
WirelessMAN-HUMAN	Point-to-multipoint	NLOS	2-11 GHz	TDD

2.2.1. Adaptive Channel Coding and Modulation

The WirelessMAN-OFDM PHY is based on OFDM modulation and designed for NLOS operation in the frequency bands below 11 GHz. It uses OFDM technology with 256-point FFT to resist delay spread, multi-path effect, and inter-symbol interference (ISI). The OFDM uses 192 data sub-carrier of 256 points for carrying data bits. The OFDM technology can help to extend transmission coverage and increase transmission data rate.

Using the OFDM technology, data bits will be modulated into OFDM symbols for transmitting after channel coding. There are four modulation scheme can be used: BPSK, QPSK, 16-QAM, and 64QAM. Using higher data-rate modulation scheme can convey more information in each OFDM symbol but harsher attenuation of signal will be accompanied.

Channel coding is composed of the following steps: randomization, Reed-Solomon code (RS), convolutional code (CC), and interleaving. In transmitting direction, each data burst should be processed and encoded in order. In receiving direction, reversed order of decoding procedures applies to the received data burst. Data randomization scrambles serial bit streams with PRBS (pseudo-random binary sequence). The Reed-Solomon code is derived from a systematic RS (N=255, K=239, T=8) code using GF(28) where

N is the number of overall bytes after encoding

K is the number of data bytes before encoding

T is the number of data bytes which can be corrected.

This code is shortened and punctured to enable variable block sizes and variable error-correction capability. The RS code is used to correct byte-level error of data

block.

The convolutional code is a punctured code and is derived from basic CC with rate of 1/2, and constraint length equal to 7. The CC is used to correct bit-level errors of bursts. These two coding scheme (RS-CC) can greatly reduce the data error rate of a radio channel.

Table 2-4 shows the defined input/output block sizes used by mandatory channel coding schemes in the standard in the order of decreasing robustness. Each block size represents the data size that can be carried in one OFDM symbol. For example, since an OFDM symbol contains 192 data sub-carrier, it can convey 24 bytes with BPSK modulation per symbol.

All encoded data bits are interleaved by a block interleaver to re-permute the bit order within a block for avoiding burst channel errors. Burst channel errors may make the Convolutional Coding work poorly.

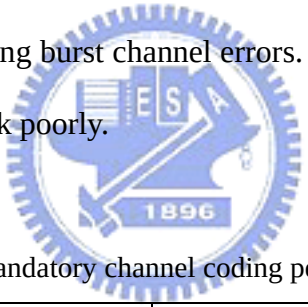

Table 2-4 Mandatory channel coding per modulation

Modulation	**Uncoded block size (bytes)**	**Coded block size (bytes)**	**Overall coding rate**	**RS code**	**CC code rate**
BPSK	12	24	1/2	(12, 12, 0)	1/2
QPSK	24	48	1/2	(32,24, 4)	2/3
QPSK	36	48	3/4	(40, 36, 2)	5/6
16-QAM	48	96	1/2	(64, 48, 8)	2/3
16-QAM	72	96	3/4	(80, 72, 4)	5/6
64-QAM	96	144	2/3	(108, 96, 6)	3/4
64-QAM	108	144	3/4	(120, 108, 6)	5/6

2.2.2. Frame Structure

The IEEE 802.16 standard uses periodical time-based frames to carry MAC

PDUs in physical layer. The frame structure used in TDD mode is illustrated in Figure 2-8. Each frame contains one downlink sub-frame and one uplink sub-frame. The compartment of downlink and uplink sub-frame is adaptive according to the schedule in current frame.

A downlink sub-frame carries one PHY PDU for base station transmitting to all other subscriber stations, while an uplink sub-frame consists of multiple PHY PDUs transmitted from different subscriber stations. Each PHY PDU is proceeded by a preamble, which is used for frame synchronization.

In the downlink PHY PDU, it contains a frame control header (FCH) burst which immediately follows the preamble. The FCH burst transmits the downlink frame prefix (DLFP) using BPSK rate 1/2 with the mandatory coding scheme. The DLFP advertises information about downlink bursts carried in downlink sub-frame.

Each data burst are transmitted with different modulation and coding scheme. Data bursts in the downlink sub-frame must appear in the order of decreasing robustness. Multiple MAC PDUs are carried in a data burst to corresponding subscriber station.

In the first burst, it uses the most robust PHY mode and contains broadcast management messages, such as DL-MAP, UL-MAP, DCD, and UCD. All subscriber stations receive the information carried in these messages to process other bursts.

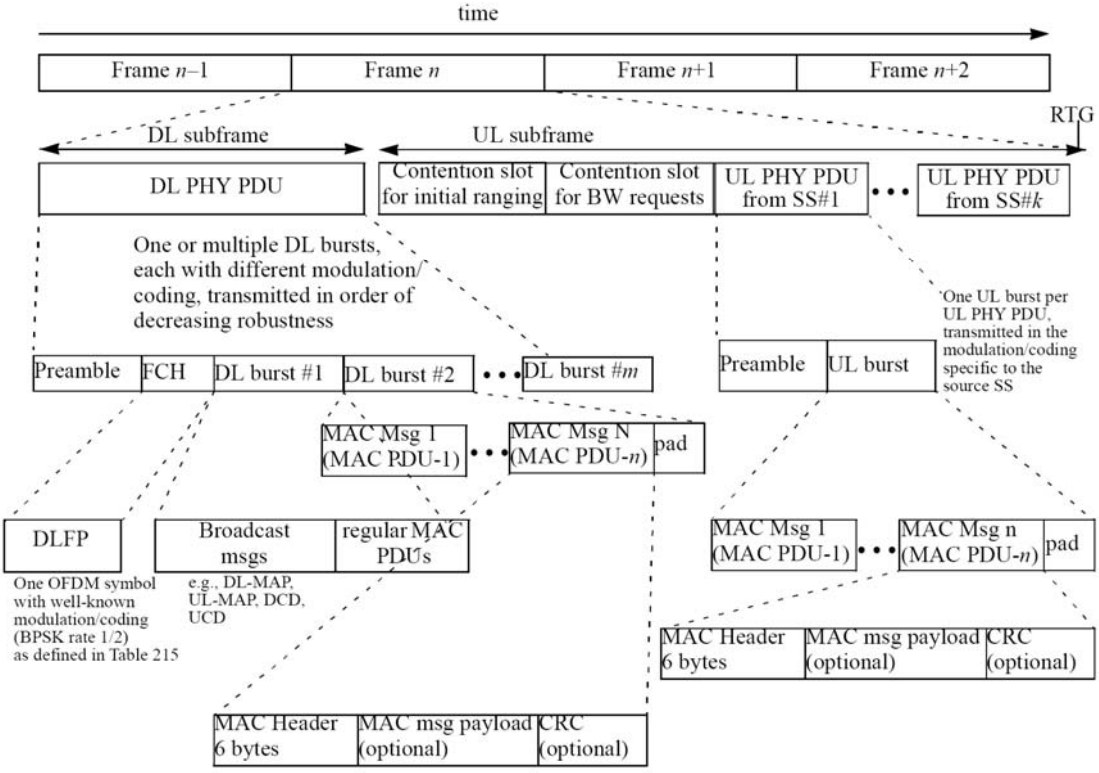

Figure 2-8 OFDM TDD frame structure

In the uplink sub-frame, it contains two contention periods for initial ranging and bandwidth request. Only in these two periods, data transmissions may encounter collisions. All other uplink PHY PDUs are scheduled by the BS and each are transmitted from different SS. Each uplink PHY PDU which is transmitted by a subscriber station contains one leading preamble and one data burst for uplink. The subscriber station transmits the uplink PHY PDU with the PHY mode and opportunity assigned in the UCD message.

Chapter 3. Design

In this chapter, we introduce the design of the module-based platform used in the NCTUns network simulator. We then describe topologies of IEEE 802.16 PMP network supported in our simulation. Finally, we explain the design of our protocol modules, including MAC layer and PHY layer modules, for NCTUns.

3.1. Module Framework in the NCTUns Network Simulator

Our work depends on the module-based platform of the NCTUns network simulator. The NCTUns network simulator is composed of the following three components: a simulation engine, a modified kernel, and a user front-end GUI program. The simulation engine program contains many protocol modules, each of which is used to simulate a specific protocol. A researcher or a module developer can easily add his/her new modules and integrate them into NCTUns, or modify existing modules to meet their needs. Besides, protocol modules can be combined or grouped to a kind of protocol stacks inside a network node.

Figure 3-1 shows an example of protocol module combinations in NCTUns. In this small network, two Host nodes are connected via a Switch node. Each Host node has an 802.3 interface. Its protocol stack is composed of an Interface module, an ARP module, a FIFO module, an 802.3 module, a PHY module, and a LINK module in sequence from the top to the bottom. For a packet transmission, the processing of layer-3 or higher-layer protocols is accomplished in kernel. After the processing in kernel, the Interface module reads an IP packet from a tunnel interface in the kernel

space. Next, it passes the packet to the module below itself to simulate the processing of protocols below layer 3. For example, the ARP module has two tasks. One is to perform the address resolution protocol that maps an IP address into a MAC address. The other is to fill in the Ethernet header with the resolved MAC address.

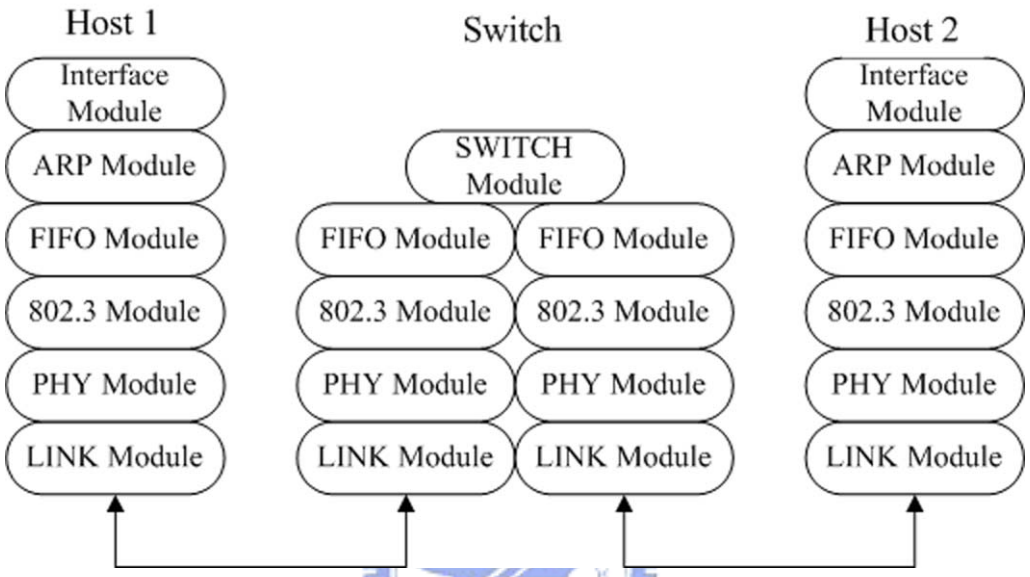

Figure 3-1 Module-based platform in the NCTUns

For a packet reception, the protocol processing starts from a layer-1 protocol. As such, the PHY module is the first one for processing incoming packets. And, the Interface module is supposed to be the final module for processing incoming packets before this packet is processed at the IP layer in kernel. The Interface module writes the received packet to a proper tunnel interface to make it be further processed by upper-layer protocols.

All traditional network nodes in NCTUns usually have a FIFO module, an 802.3 module, a PHY module, and a LINK module. A FIFO module buffers packets into a transmission queue in the first-in-first-out order to avoid dropping packets that cannot be processed immediately. The 802.3 module handles the 802.3 medium access control layer functionalities, including frame transmission scheduling, received frame

processing, and retransmissions due to collisions. The PHY module is used for simulating the characteristics of Ethernet physical layer, such as bit error, and propagation delay. The LINK module is created for specifying the connectivity of a node in a simulated network.

The NCTUns network simulator is written in C++ and provides a basic prototype module NslObject. Every protocol module in NCTUns should inherit from this prototype module and substitute some primitive member functions by their own. Figure 3-2 shows the declaration of NslObject class. The most important member functions of this module are the *send()* and *recv()* functions. When a node receives a packet from other nodes, the *recv()* member function of the bottom module (LINK) is called first. Then, the *recv()* member function of upper modules will be called in the bottom-to-top manner if needed. Contrary to the *recv()* member function, the *send()* member function processes packets sent from the node itself to other nodes. When a packet is sent, the *send()* function of the topmost module (INTERFACE) will be called first to process it. Then, the packet is passed to the *send()* function of next lower-level module. This procedure will repeat until the packet is transmitted out or is dropped by some module. Figure 3-3 shows the flow sequences of the sending and receiving procedure in NCTUns.

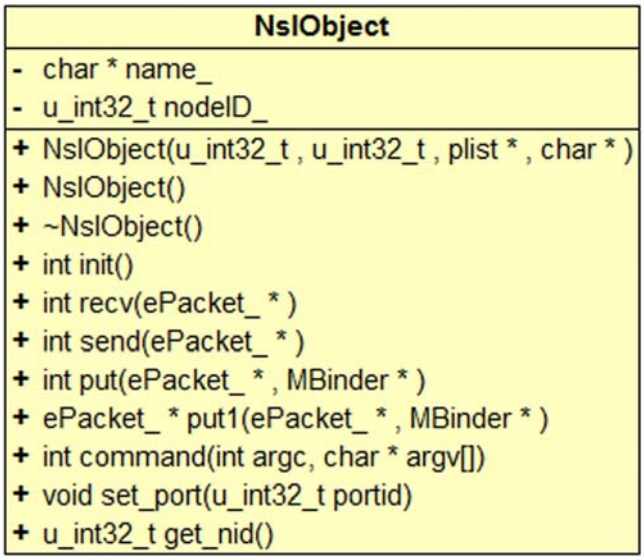

Figure 3-2 NslObject

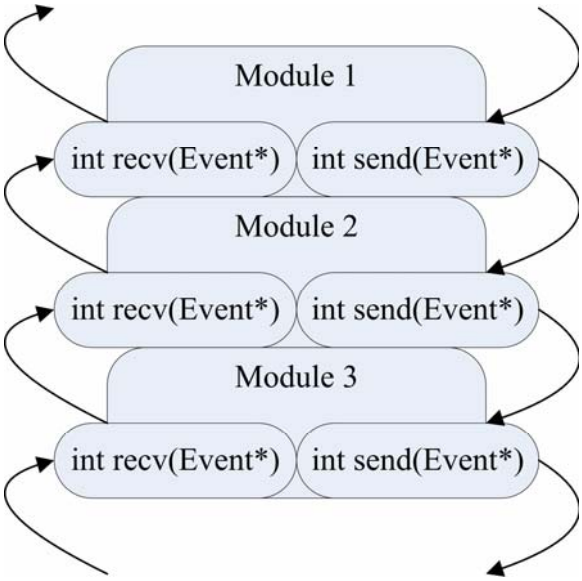

Figure 3-3 Packet flow architecture within modules

3.2. Supported Network Topologies and Usages

To support 802.16 PMP networks on the NCTUns network simulator, we add three new types of nodes, which are called BS Gateway, SS Client, and SS Gateway, respectively.

Figure 3-4 Icons of three new nodes of WiMAX PMP networks

The BS Gateway is an 802.16 base station with two interfaces. One is 802.3 Ethernet interface and the other is 802.16 PMP-mode BS interface. The 802.3 interface connects to Internet with a wired link. The 802.16 interface is an 802.16 PMP-mode radio to communicate with neighboring 802.16 subscriber stations (SS). In addition, a BS node is responsible for arranging the transmission schedule in an 802.16 network.

The SS Client node is one type of 802.16 subscriber station. It is like a notebook equipped with an 802.16 wireless network interface card (NIC). This type of nodes can attach to an 802.16 network via broadband.

The SS Gateway node is another kind of subscriber station. Like a gateway, it has two interfaces, including one 802.3 wired interface and one 802.16 wireless interface. The 802.3 interface connects to a small LAN or a private network and the 802.16 interface is used to communicate with 802.16 base stations. Hosts in a LAN or a private network dominated by an SS Gateway can access Internet through the SS Gateway. In such situation, the SS Gateway acts as a wireless gateway.

We adopted the IP-CS scheme to bridge our WiMAX protocol modules and traditional protocol modules. The IP-CS scheme converts IP packets into MAC PDUs at CS sub-layer. Therefore, MAC PDU encapsulates packet as payload IP and it can be transmitted to a WiMAX network. Three basic 802.16 network topologies supported by NCTUns are shown in Figure 3-5, and the protocol layer architectures

used in our design can be presented as Figure 3-6. More complex network topologies can be built by mixing these three basic networks and other supported networks currently supported by NCTUns.

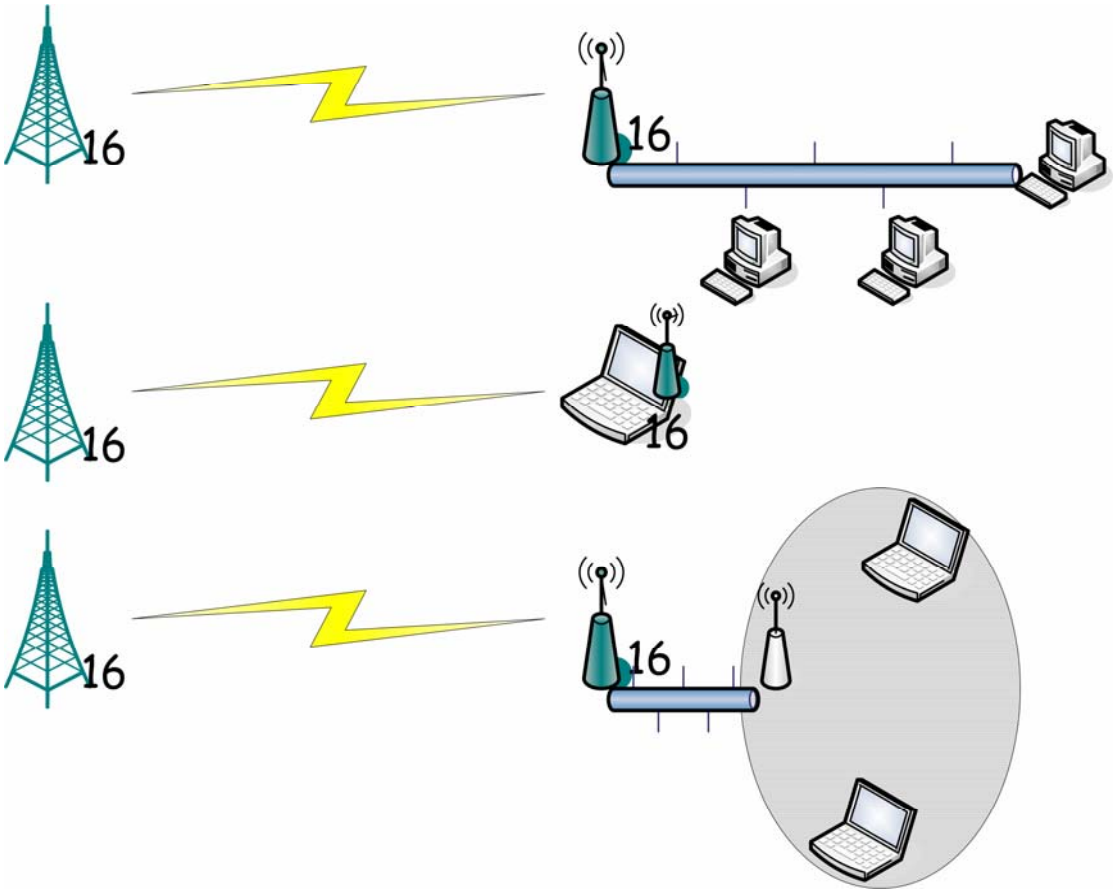

Figure 3-5 Sample topologies of WiMAX PMP-mode network

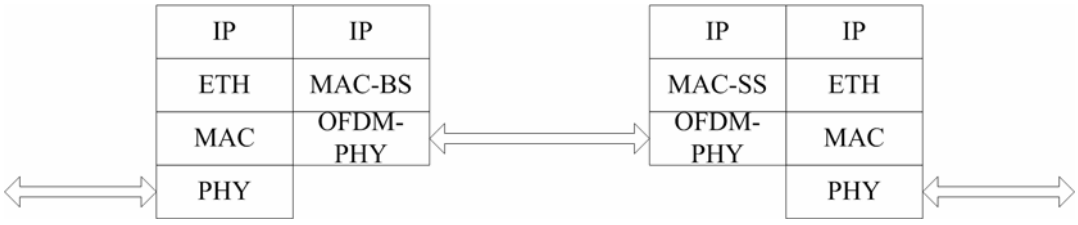

Figure 3-6 Protocol Layer Architecture for IP-CS

3.3. Protocol Stacks of 802.16 Components in NCTUns

3.3.1. BS Gateway Node

An 802.16 BS Gateway node is evolved from a router node in NCTUns. It has two interfaces in our design. One is an 802.3 Ethernet interface, and the other is an 802.16 radio. It acts as a gateway for other subscriber stations to connect to the Internet. Its 802.3 Ethernet interface must have enough bandwidth to support entire WiMAX network. This 802.3 Ethernet interface uses the 802.3 protocol stack modules (Interface, ARP, FIFO, MAC8023, and PHY modules) already supported in NCTUns. The 802.16 interface needs to work in accordance with the IEEE 802.16 standard. In our implementation, an 802.16 interface has the following modules: an Interface module, a MAC802_16_PMPBS module, an OFDM_PMPBS module, and a LINK module.

The MAC802_16_PMPBS module is responsible for the functionality of the MAC layer of an 802.16 base station, and the OFDM_PMPBS module performs physical layer functions. There are several special designs in our implementation to reduce the design complexity. First, our WiMAX modules carry IP packets instead of Ethernet frames, so we do not need ARP module to fill up the Ethernet header. Second, because the transmission queue management is accomplished in the MAC802_16_PMPBS module, the FIFO module, used for interface queue management, is not necessary for our 802.16 protocol stack.

When the BS Gateway receives packet, the packet will be send into kernel. Then, the kernel will route the packet to proper interface. That is, the packet will be routed to the Interface module belonging to an 802.16 protocol stack of this BS Gateway if

the packet is destined to a subscriber station. Next, the packet will be sent to the MAC802_16_PMPBS module. The MAC802_16_PMPBS module decides which subscriber station this packet should be transmitted to and pushes it into connection queue. The packet will be transmitted to other subscriber stations after scheduling.

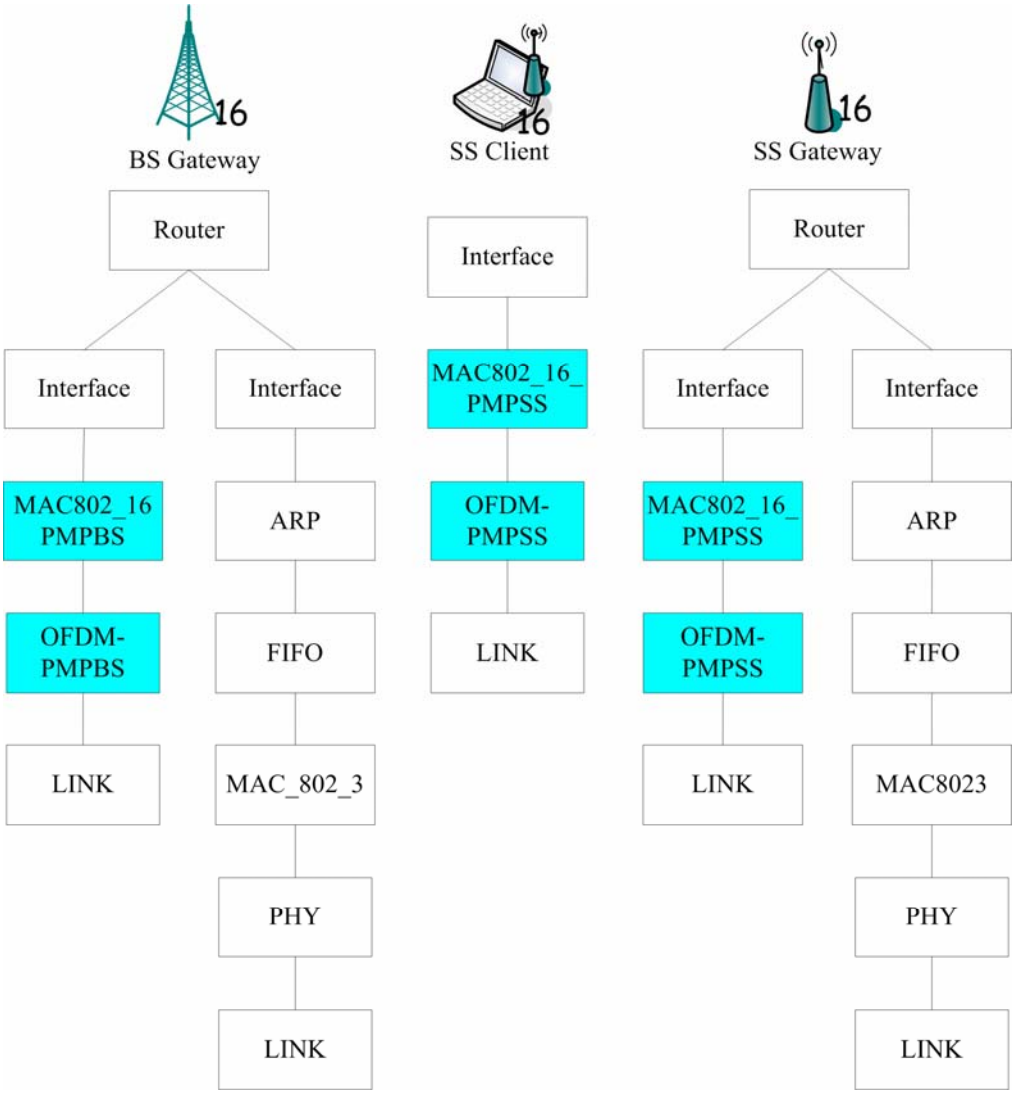

Figure 3-7 Protocol stacks of 802.16 components in NCTUns

3.3.2. SS Client Node

An SS Client node acts as a subscriber station of WiMAX network. Its protocol stack contains a MAC802_16_PMPSS module and an OFDM_PMPSS module. The MAC802_16_PMPSS module performs MAC layer functionalities of a subscriber

station and the OFDM_PMPSS module performs the functionalities of the physical layer. Since the functions of the MAC layers and PHY layers between SS and BS are quite different, each of term is implemented as a separate protocol module, respectively.

3.3.3. SS Gateway Node

An SS Gateway node has a similar protocol stack to a BS Gateway node. It also has one 802.16 interface and one 802.3 interface to exchange packets between two different networks (802.3 and WiMAX networks). The 802.16 interface has the same protocol stack (MAC802_16_PMPSS and OFDM_PMPSS module) as an SS Client. Briefly, an SS gateway forwards data packets received from a base station to its backend network and forwards data packets from its backend network to a base station. Contrarily, since a subscriber station only connects with a base station, it simply transmits all packets to the base station.

3.4. Design of WiMAX modules

3.4.1. MAC Layer Modules

3.4.1.1. MAC Layer Function

In our design, the functionalities and procedures of the 802.16 MAC layer can be partitioned into the following parts: network entry procedure, management message negotiation, network management, connection management, and transmission scheduling.

A BS node broadcasts the network information, such as DCD and UCD messages, periodically. When a SS is trying to attach to a WiMAX network, it must

do the network entry procedure to get network information from the BS and notify the BS node of its communication capability. The network entry procedure exchanges many management messages, which are used for preparing connection establishment between a BS and an SS. Our implementation conforms to this mechanism defined in the standard to exchange the required parameters between BS and SS nodes; however, some exchanged information is optional and not necessary in NCTUns.

For example, the IP connectivity description and the operational parameters of the physical layer are not needed. In our implementation, unnecessary messages are ignored to reduce the program complexity without affecting the accuracy of simulation results.

When subscriber stations perform the network entry procedure, the base station that dominates this network also controls the network utilization and assigns the connection identifications to these new subscriber stations. In this phase, the BS may perform an admission control mechanism to control the network utilization. The rules of the admission control mechanism may be defined case by case for different situations and goals. Although we have provided some basic quality-of-service (QoS) rules in our simulation cases, we do not enable the QoS feature at present, but leave it as a topic for future work.

After the network entry procedure, each SS will establish a basic connection, a primary management connection, and a data transport connection with the BS. The basic and primary management connections are used for management and control purposes. The data transport connection is only used for transmitting data packets. Data packets sent from upper-layer protocols will be classified into different connections and wait for scheduling.

The operational principle of the 802.16 PMP-mode is that all data transmissions in such a network should be controlled by a BS. Namely, only the BS dominating this

network has the right to schedule the transmissions of data packets. There are two advantages using this principle. First, the network bandwidth can be used effectively. Packet collisions will not occur using this principle. Second, the quality-of-service can be guaranteed easily.

When the Interface module sends a IP packet, *send()* function will be called. Then the MAC module will perform a simple classification. The classifier finds an appropriate connection to transmit the IP packet according to the destination IP address.

In our implementation, a MAC-layer protocol module is composed of two parts, MAC CS and MAC CPS. The MAC CS classifies packets and maps them into different connections. A connection encapsulates classified packets into MAC PDUs in MAC CPS. In MAC CPS, the Cyclic Redundancy Check (CRC) is used for checking the correctness of data payload. Although the RS-CC coding in the PHY module can correct burst errors over an error-prone wireless channel, it does not guarantee to correct all errors. If the MAC PDU fails in the CRC checking, it will be dropped.

PDU Fragmentation is helpful for utilizing the bandwidth effectively and avoiding waste of bandwidth. Without using the PDU fragmentation mechanism, the MAC layer has two choices when there is a packet, the length of which is greater than the maximum scheduled transmission length. First, we can drop this large packet. Second, we can wait for next opportunity to transmit the packet. The former may reduce the throughput performance of user-level programs significantly, and the latter may cause the head-of-line blocking problem.

In contrast, by using the PDU fragmentation mechanism, a large MAC PDU can be partitioned into several smaller fragments with the same coding and modulation combination. As such, data packets can be transmitted without waiting for a

transmission opportunity that is large enough to carry the original large PDU.

Furthermore, these smaller fragments can be concatenated into a data burst, which is variable-length transmission unit. A data burst is helpful for a node to better utilize network bandwidth. Consider the case that a SS node is assigned a very large transmission opportunity. Without the aid of bursts, the SS node can send only one PDU at each transmission opportunity even though the transmission opportunity is large enough to carry two or more PDUs. This will lead to under-utilization of the allocated bandwidth because the SS node cannot fully utilize its assigned opportunities. Using bursts, the SS node is able to group several small PDUs into a burst, to generate a length nearly large as that of the transmission opportunity. In this way, the bandwidth utilization for a node will approach its allocated transmission opportunities.

Note that the length of a data burst must be multiples of block size. Dummy data will be padded at the tail of a data burst if we cannot find PDUs to form a burst whose length is a multiple of a block. We do not implement the ARQ sub-header and the packing sub-header mechanisms since the ARQ sub-header is optional in the standard and the packing sub-header mechanism is only useful in ATM networks.

The transmission scheduling algorithm of a BS node is quite different from that of an SS. In the following sections, we will introduce how they work, respectively.

3.4.1.2. BS MAC Layer Module

In a WiMAX PMP network, a BS is responsible for scheduling data transmissions of all network nodes and the management of resources.

The BS communicates with subscriber stations through connections. Each connection works as a queue and has its unique connection identification (CID).

Based on connection identifications, connections can be classified into three types in our design.

The first type is used for delivering general management messages between the BS and a particular SS. For example, an SS will be assigned a basic connection, a primary management connection, and a secondary management connection after it attaches (registers) to the network successfully. The second type is only used for processing data packets.

The third type of connections are not owned or assigned to any SS nodes. They include the broadcast connection and the initial ranging connection. The broadcast connection is only used by the BS for transmitting broadcast information to all SS nodes. The initial ranging connection is used during the network entry phase for an un-registered SS node to attach to this network. Since these global connections are never assigned to any SS nodes, they use pre-defined CIDs, which are well-known by all nodes.

During the network entry phase, the operations can be divided into a sequence of stages. At this phase, the BS node has to keep track of the statuses of SS nodes that are performing the attaching procedure to know their registration progresses. After the network entry phase, the BS also records the configurations and settings of each registered SS node, which will be used for management purposes. In our implementation, a C++ class, “SS Object”, in a BS node represents the entity that keeps track of the information of a remote SS node.

Each SS node has two mandatory management connections, one basic connection and one primary connection. In addition, a SS node may either have multiple data transport connections or none of them. In other words, every CID except those global used is owned by a particular SS and can represent a particular SS. Therefore, the BS can delegate the SS Object to handle its owned connections.

However, the global connections, especially initial connection, are not owned by any SS, the BS must handle these connections by itself. Figure 3-8 illustrates the relations between connections and network nodes. Small circles (send, recv, and classifier) denote a simple procedure which only examines the input PDU and passes it to the next procedure after the PDU passes the examination. Another type of circles (BS, SS, and SS Object) denotes a more complex procedure which not only examines the received PDUs (in this case, those PDUs are management messages) but also responds a message to peer.

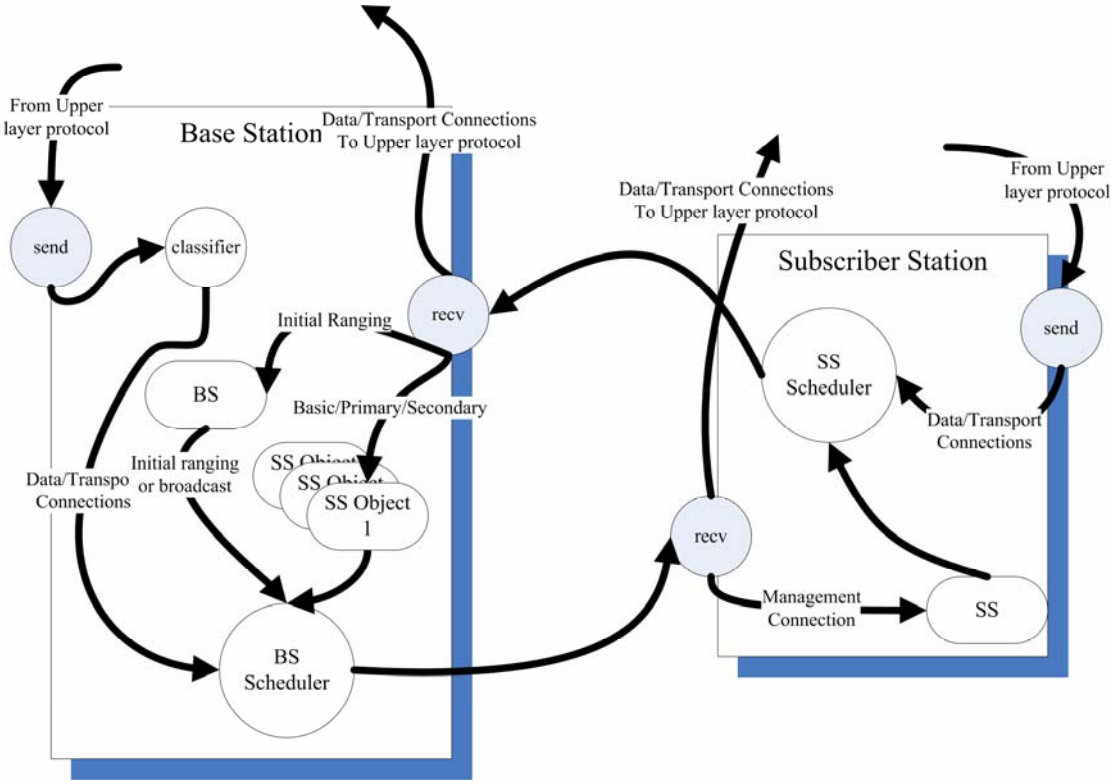

Figure 3-8 Architectures of MAC layer

As shown in Figure 3-8, the BS MAC layer may receive packets from either an upper layer protocol or other SS nodes. If a packet transmitted from the upper-layer protocols reaches the BS MAC layer, the send procedure will be invoked to process this packet. In contrast, if a packet from an SS node reaches the BS node, it will trigger the recv() procedure to process the incoming packet. The packet will be passed

to upper layer for routing. If the packet is destined to other SS, then it will re-enter into the MAC module.

The BS scheduler is responsible for deciding how to transmit data packets based on the physical layer specification. In such a condition, the scheduler should be able to know the operational information about the physical layer protocol. The BS scheduler calculates downlink and uplink schedules for SS nodes on the frame-by-frame basis.

In an uplink schedule, the BS scheduler always assigns a constant number of symbols for every frame by the bit rate configured by users. This approach is like the UGS schedule. In a downlink schedule, since the modulation/coding mode used by a connection is determined by SS nodes, the BS scheduler checks every connection in the round-robin manner and allocates the number of packets as fair as possible. The schedule principle used by the BS scheduler here is trying to fill all transmission opportunities with fragments as compactly as possible.

According to the results of uplink and downlink schedules, the BS scheduler generates the DL-MAP and UL-MAP messages, which will be sent over the broadcast connection. The scheduling procedure also illustrated in Figure 3-9 and Figure 3-10.

The DL-MAP and UL-MAP messages are carried in all of control frames transmitted over the broadcast connection. A control frame is made up of a broadcast message, a DL sub-frame, and a UL sub-frame in sequence. However for the convenience of implementation, the calculation of those components should be carried out in reverse order. To avoid that the DL and UL sub-frames run out of the space of the control frame, the BS scheduler reserves minimal space for the broadcast message before it starts to calculate the schedule of DL and UL transmissions. Finally, the BS scheduler generates a list of bursts in the order sorted by the robustness of modulation/coding combinations used by those bursts.

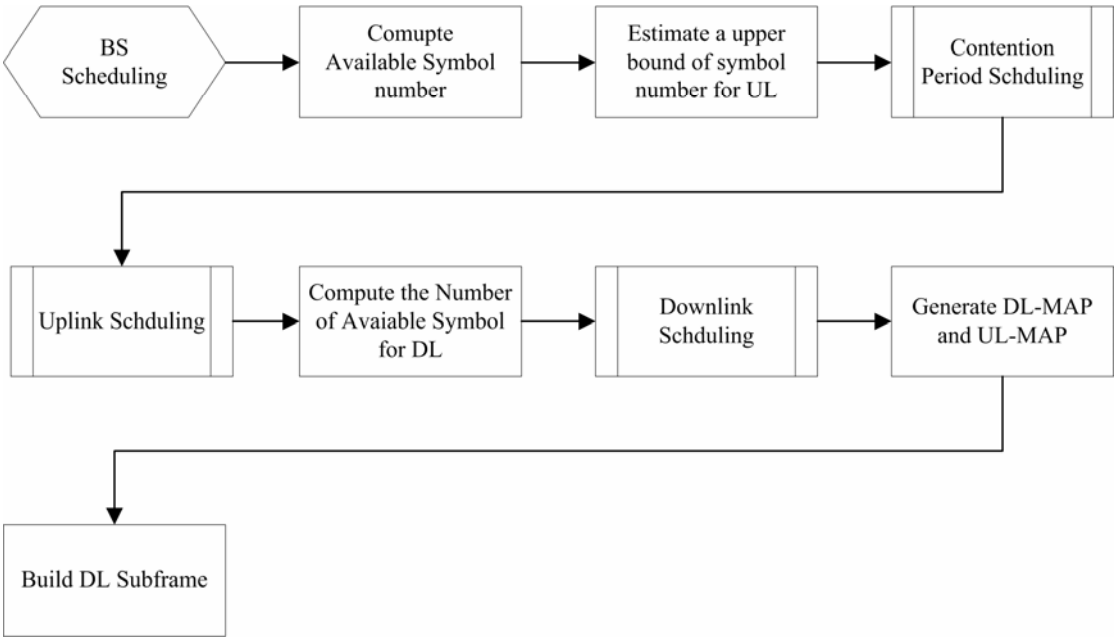

Figure 3-9 BS Scheduling

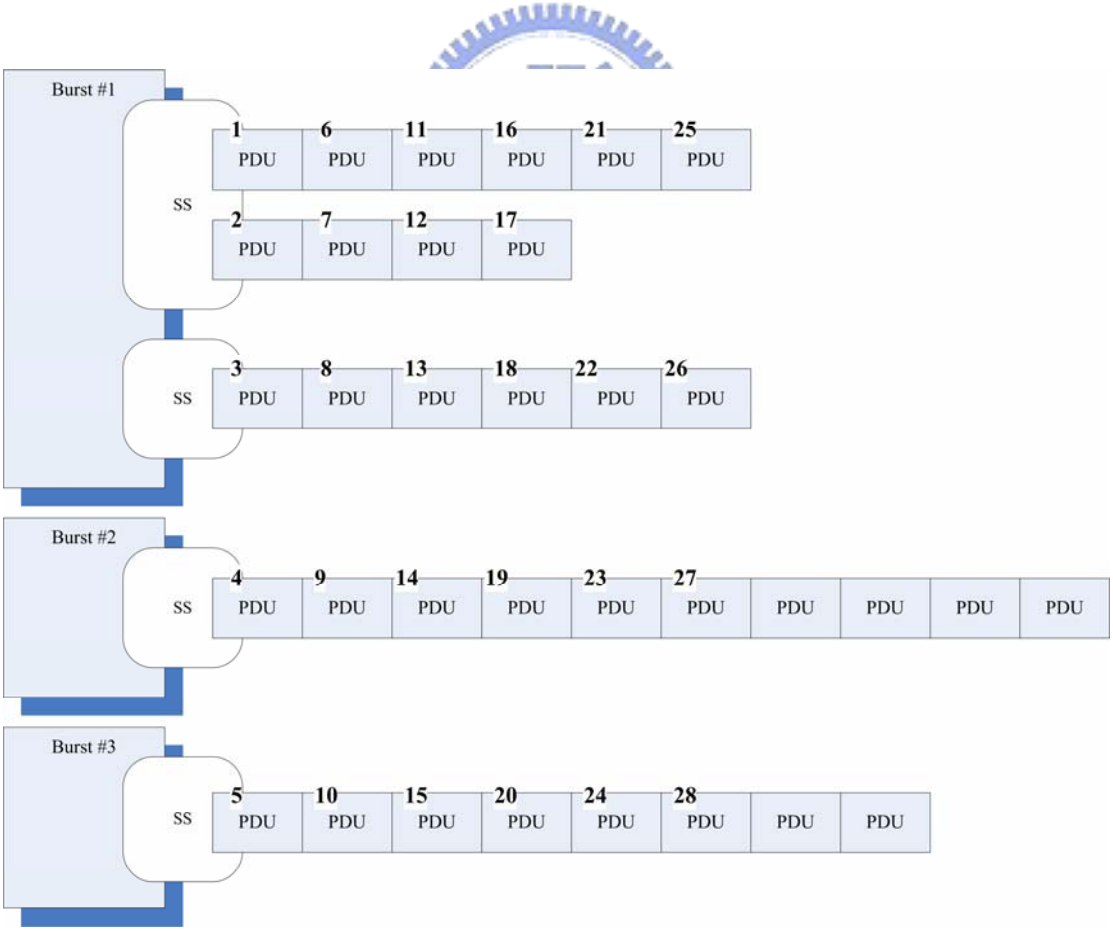

Figure 3-10 Transmission Scheduling

3.4.1.3. SS MAC Layer Module

Unlike the BS MAC module, the SS MAC module is simple and has fewer duties. The SS MAC module executes the uplink transmission at scheduled times indicated in the UL-MAP assigned by the BS scheduler. That is to say, the SS MAC module just schedules its connection queues for uplink schedule. Note that the SS MAC module uses the same transmission scheduling illustrated in Figure 3-10 as BS.

3.4.2. Physical Layer Modules

3.4.2.1. Channel Model and Channel Coding

In the physical layer, we don't only simulate the functionalities, but also perform a wireless environment model. It includes two major tasks, channel model and channel coding. We adopt the channel path loss model presented in [1] [2]. It presents a path loss model based on a statistical approach using the experimental results collected in the real world is presented. The path loss model was formulated with a Gaussian random variation. This model also takes the shadow fading and slow fading effects into account. In the paper [1], it also proposed three terrain category parameters listed in Table 3-1. We use the terrain category B in our works.

Table 3-1 Parameters for pass loss model

MODEL PARAMETER	TERRAIN CATEGORY		
	A (Hilly/Moderate-to-Heavy Tree Density)	B (Hilly/Light Tree Density or Flat/Moderate-to-Heavy Tree Density	C (Flat/Light Tree Density)
a	4.6	4.0	3.6
b	0.0075	0.0065	0.0050
c	12.6	17.1	20.0
σ_γ	0.57	0.75	0.59
μ_σ	10.6	9.6	8.2
σ_σ	2.3	3.0	1.6

We compute signal-to-noise ratio (SNR) based on the transmit power and variant path loss and noise. Then we derive the uncoded bit error rate (BER) from the signal-to-noise ratio (SNR) and energy-to-noise ratio (ENR) for each different modulations BPSK, QPSK [3], 16QAM, and 64QAM [4]. These modulations are all over Rayleigh fading instead of AWGN (Additional White Gaussian Noise). After calculating of BER, we apply errors to simulate the wireless channel environment by randomly choosing $BER * BurstLength$ bits in the burst.

$$
\begin{aligned}
P_{RX} &= P_{TX} - L_F \\
SNR &= P_{RX} - N \\
ENR &= \frac{Signal / dataRate}{k * T} \\
&= \frac{Signal / dataRate}{Noise / bandwidth} \\
&= \frac{Signal / Noise}{dataRate / bandwidth} \\
&= SNR - 10 * \log(\frac{dataRate}{bw}) \\
BER &= computeBER(ENR, modulation)
\end{aligned}
$$

And BER depends on the modulation:

$$BPSK/QPSK: \frac{1}{2(1+ENR)}$$

$$M-QAM: 2(\frac{\sqrt{M}-1}{\sqrt{M}})\frac{1}{\log_2 M}\sum_{i=1}^{\sqrt{M}/2}(1-\sqrt{\frac{1.5*(2i-1)*ENR*\log_2 M}{M-1+1.5*(2i-1)*ENR*\log_2 M}})$$

The resulted curves of above channel models are shown in the following figures. In Figure 3-11, the relationship between SNR and uncoded BER is plotted, while in Figure 3-12 and Figure 3-13, the curves that represent ENR and distance versus uncoded BER are shown, respectively. Because QPSK can be regarded as a pair of orthogonal BPSK systems, BPSK and QPSK have almost equivalent curves under the same ENR.

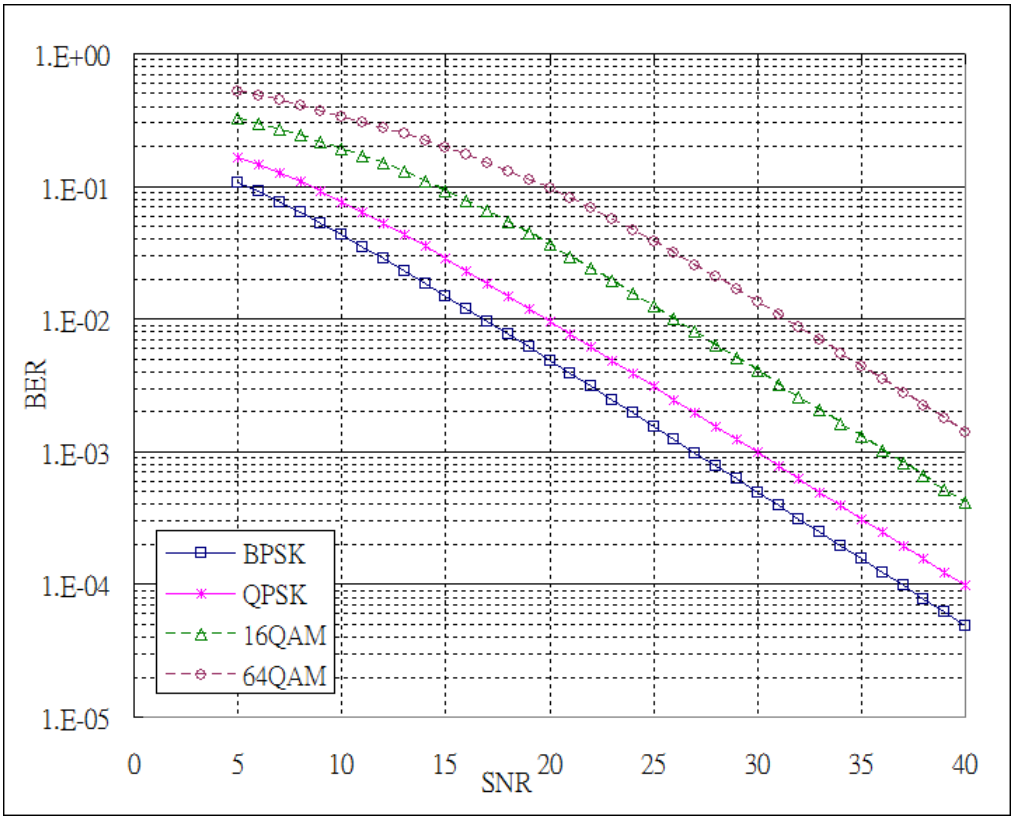

Figure 3-11 SNR vs. uncoded BER

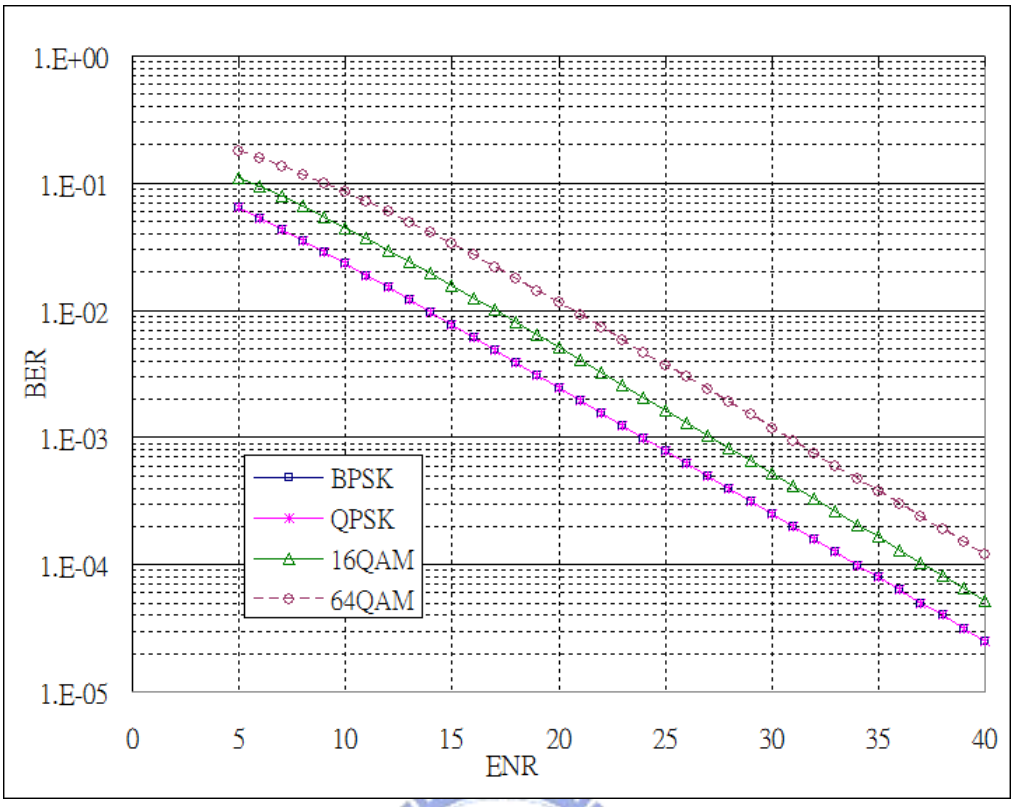

Figure 3-12 ENR vs. uncoded BER

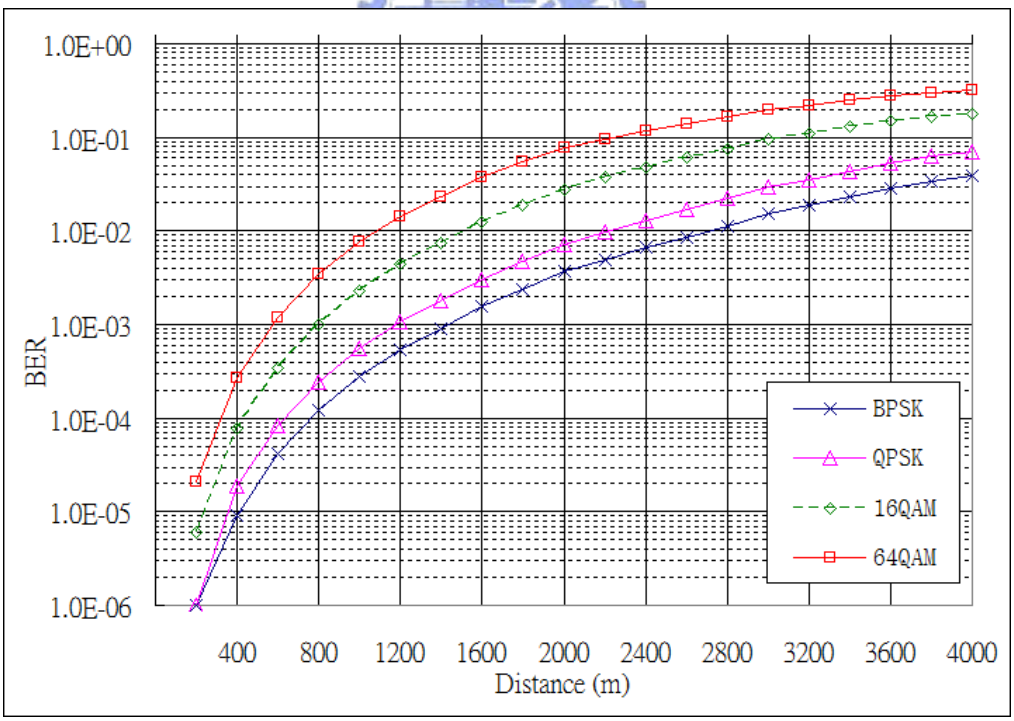

Figure 3-13 Cell distance vs. uncoded BER, Pt = 37dBm

The Reed-Solomon Code and the Convolutional Code is two major components

of Channel coding used in the 802.16 standard. The encoding of the Convolutional code is ordinary and we use the Viterbi algorithm for Convolutional decoding. With regard to Reed-Solomon Code, we used a Reed-Solomon Code library [5] for encoding and decoding.

3.4.2.2. BS Physical Layer Module

The BS activates frames in the network and transmits downlink sub-frames at the beginning of each frame duration time. The OFDM downlink sub-frame starts with a long preamble and followed by a DLFP in FCH. The BS physical layer module should build this DLFP information based on the current frame. In the base station, its MAC layer builds the schedule of current frame, and passes the schedule and data to physical layer. Next, the physical layer builds the needed information, such as DLFP, and encodes these data into burst, then transmits these data bursts to all subscriber stations.

3.4.2.3. SS Physical Layer Module

The SS physical layer module receives downlink sub-frames and examines the DLFP to determine how to decode the succeeding bursts. The MAC layer may also order the physical layer to decode a particular burst according to the DL-MAP information. The SS physical layer module transmits uplink bursts with a long preamble at scheduled times specified in UL-MAP. The MAC layer will activate the physical layer at an appropriate time.

Chapter 4. Implementation

In this chapter, we present the main idea and architecture of implementation for supporting 802.16 PMP-mode on the NCTUns network simulator. We implement the functionalities based on the Standard IEEE 802.16-2004. The protocol modules in our implementation contain MAC layer modules and physical layer modules. These two layers both have one module for the BS node (MAC802_16_PMPBS module and OFDM_PMPBS module) and one module for the SS node (MAC802_16_PMPSS module and OFDM_PMPSS module).

4.1. The Architecture of MAC Module

In this section, we introduce some important classes of the MAC layer protocol modules and show its class diagram to illustrate the relations between each class.

4.1.1. Class Diagram

There are one *mac802_16_PMPBS class* and one *mac802_16_PMPSS class* corresponding to the operation of the BS and the SS MAC layer module, respectively. Both two classes which are used as an NCTUns module shall be derivations of *NslObject class*. In the *mac802_16_PMPBS class*, it maintains a list of SS Object of *ssObject class* to assist the BS node managing the corresponding SS node. The *ssObject class* keeps track of the information of a remote SS node.

Since all the 802.16 MAC modules have many common attributes and operations, we append a *mac802_16 class* which inherits from *NslObject class* as base class for all 802.16 protocol modules. The *mac802_16 class* contains most of basic attributes

and common operations which can be shared by the BS node and the SS node. The *mac802_16 class* is used for basic prototype for WiMAX networks, not only in PMP-mode but also in Mesh mode. Similarly, we extract the same attributes between *ssObject class* and *mac802_16_PMPSS class* into *mac802_16SSbase class* since they also have some common attributes. The relations between these classes are shown as Figure 4-1.

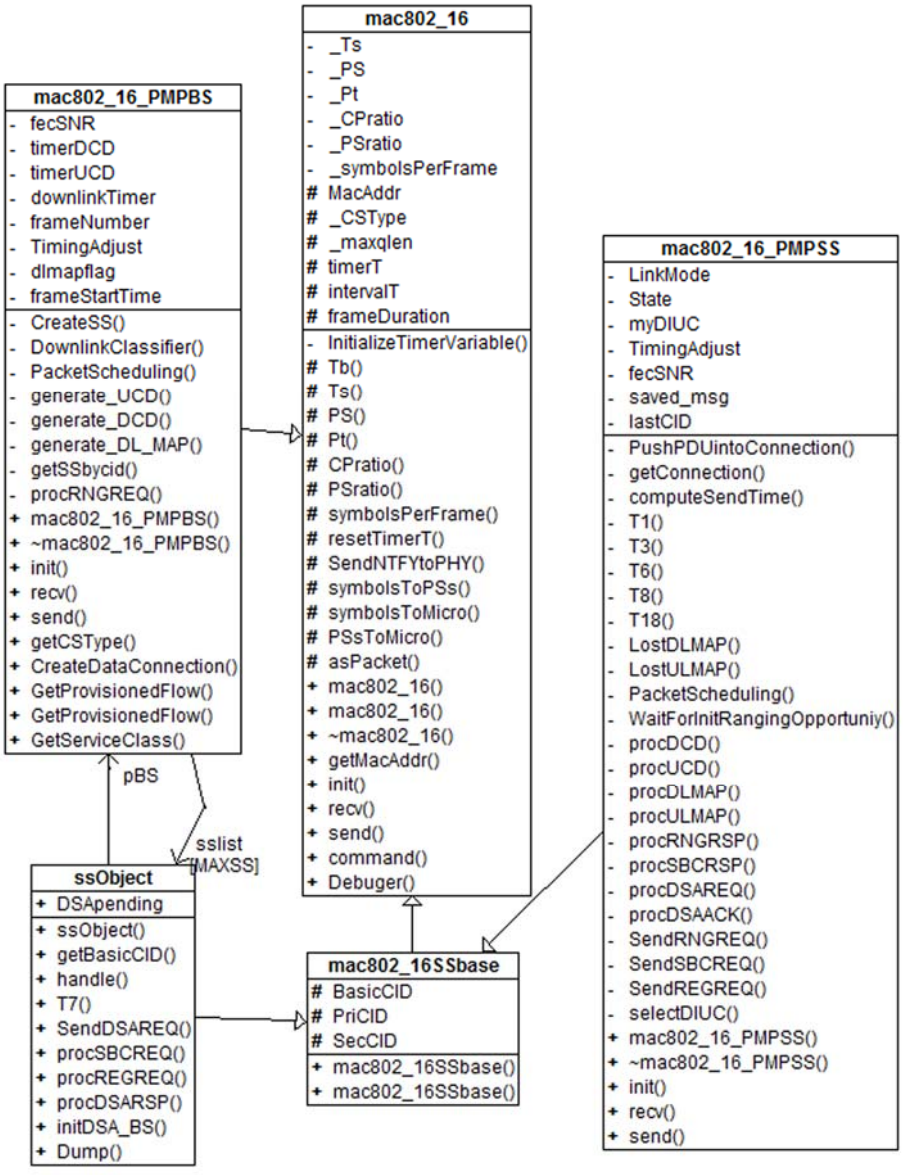

Figure 4-1 Class Diagram of MAC Layer Modules for WiMax PMP-mode

Figure 4-2 shows the relation between the necessary MAC module and its

scheduler. The MAC modules have their own schedulers to do scheduling, and they declare the scheduler as their friend class. Therefore, the scheduler can access all members of MAC module to get the needed information and connection queue status directly. We separate the scheduler as a new object in order to split schedule policy from the MAC functionalities. We hope this approach can help us to attain the flexibility of scheduling.

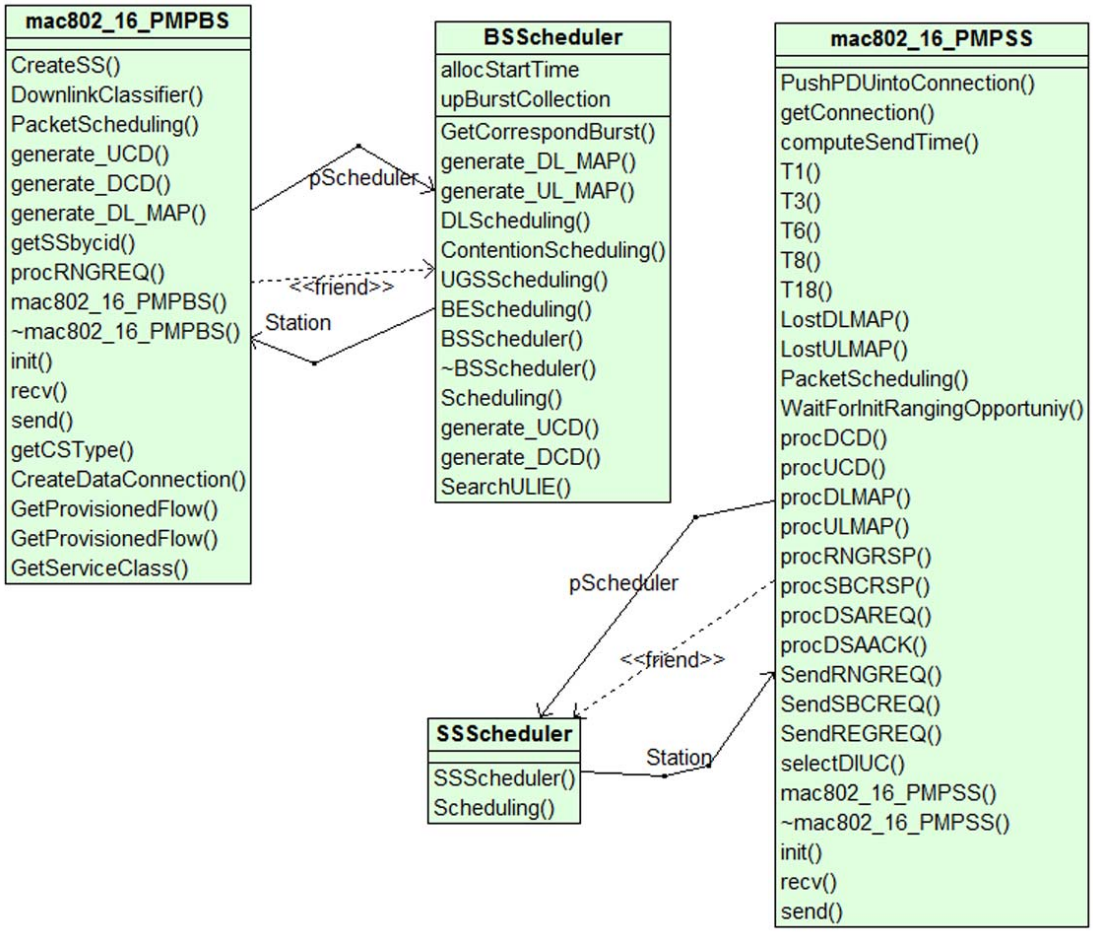

Figure 4-2 BS and SS scheduler

For supporting the MAC management message format depicted in IEEE 802.16 standard, we use *ManagementMessge class* to process the management message. All management messages have a type field at first byte and follow with variable length payload. Since the payload format varies from message to message, we need to

develop an applicable data structure for each message. There are two approaches to achieve it. We can define either a new class for each message to process or a general class to process all messages. We observe that most management messages format used in PMP-mode are as Figure 4-3. The management message payload starts with multiple fixed length fields and then variable length TLV encoding are followed.

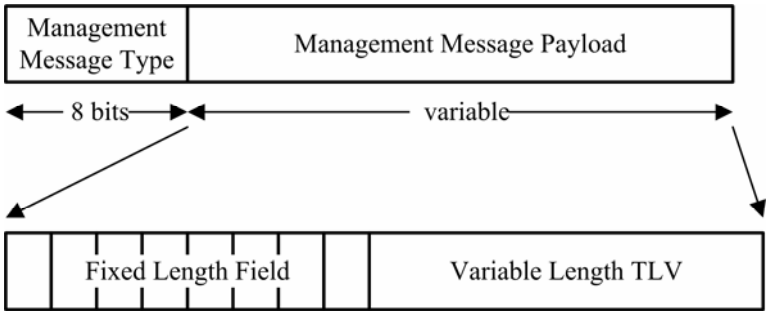

Figure 4-3 MAC Management Message Format

Each message may contain different format of fields and TLV encodings in its payload from others. Therefore, we use an *ifmgmt class* which inherits *ManagementMessage class* to process these general managements. The *ifmgmt class* contains a buffer space to store fixed length field and a pointer to *ifTLV class* used to process variable length TLV encoding. The *ifmgmt class* can take a *pFieldSize* as parameter to set its field size. It contains two virtual functions: *copyTo()* and *getLen()*. The function *copyTo()* serializes entire message into character array and the function *getLen()* helps to measure required space for serialization. The *ifmgmt class* and *ifTLV class* allocate memory spaces to store the message information when the MAC module raises a new message. They also take existing buffer via pointers to interpret the buffer into management messages when the MAC module receives a MAC PDU from lower physical layer module. In Figure 4-4, we show the declarations and relations of *ManagementMessage class*, *ifmgmt class*, and *ifTLV class*.

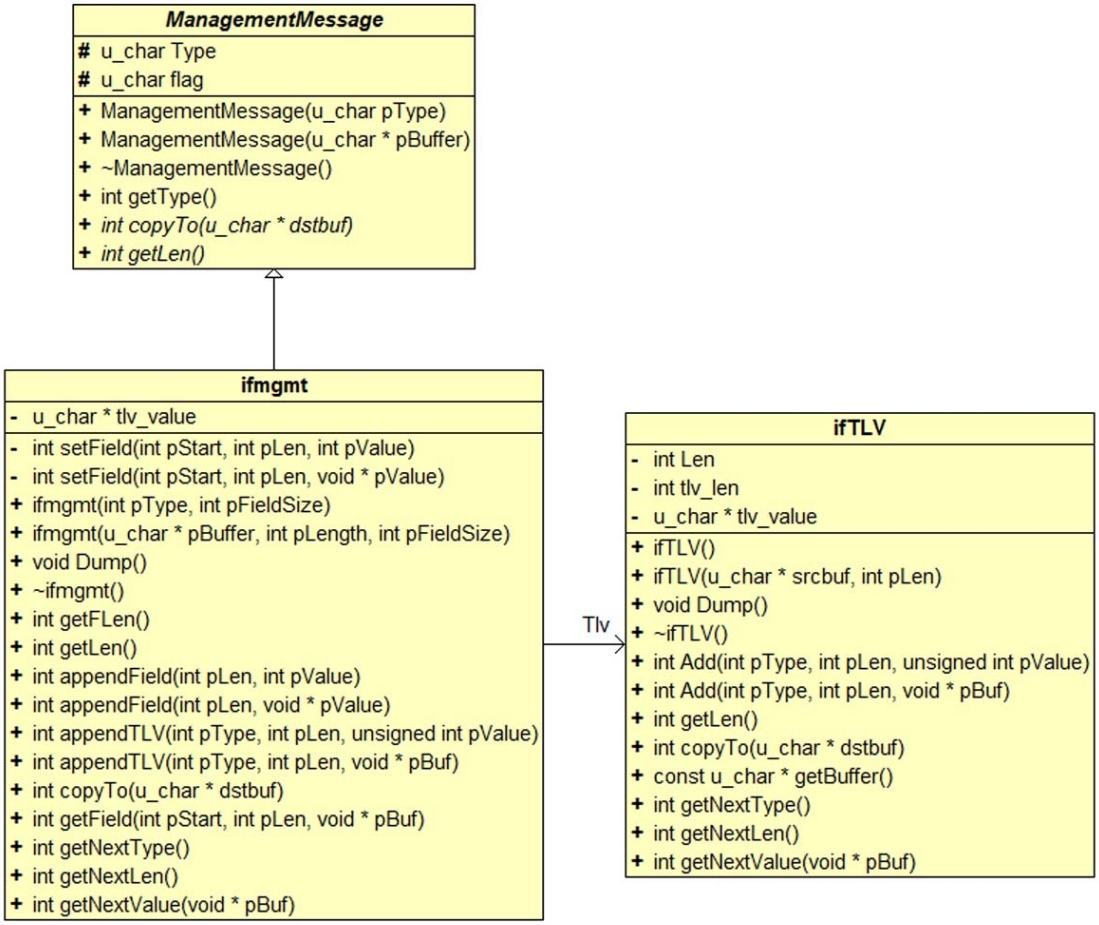

Figure 4-4 Class Diagram for Processing Management Messages Objects

All communications in 802.16 MAC layer are connection basis. Each connection with different CID can respectively perform the operations of QoS, fragmentation, packing, CRC checking and even ARQ to transmit PDUs. We design all kinds of connection as a separated hierarchy to retain ease-maintenance and extensible capability. Figure 4-5 shows the hierarchy of all connections. All kinds of connections are based on the abstract class: *Connection* to inherit its features such as the connection identification (*DataCID*), a packet queue (*PduQ*) used to store packet waiting for transmit, and other basic operations. The Connection class has two pure virtual function, *GetInformation()* which reports packet queue information and *EncapsulateAll()* which encapsulates packets in the queue as PDU payload into MAC

PDU. All classes inherited from the *Connection class* must implement these two functions of its own.

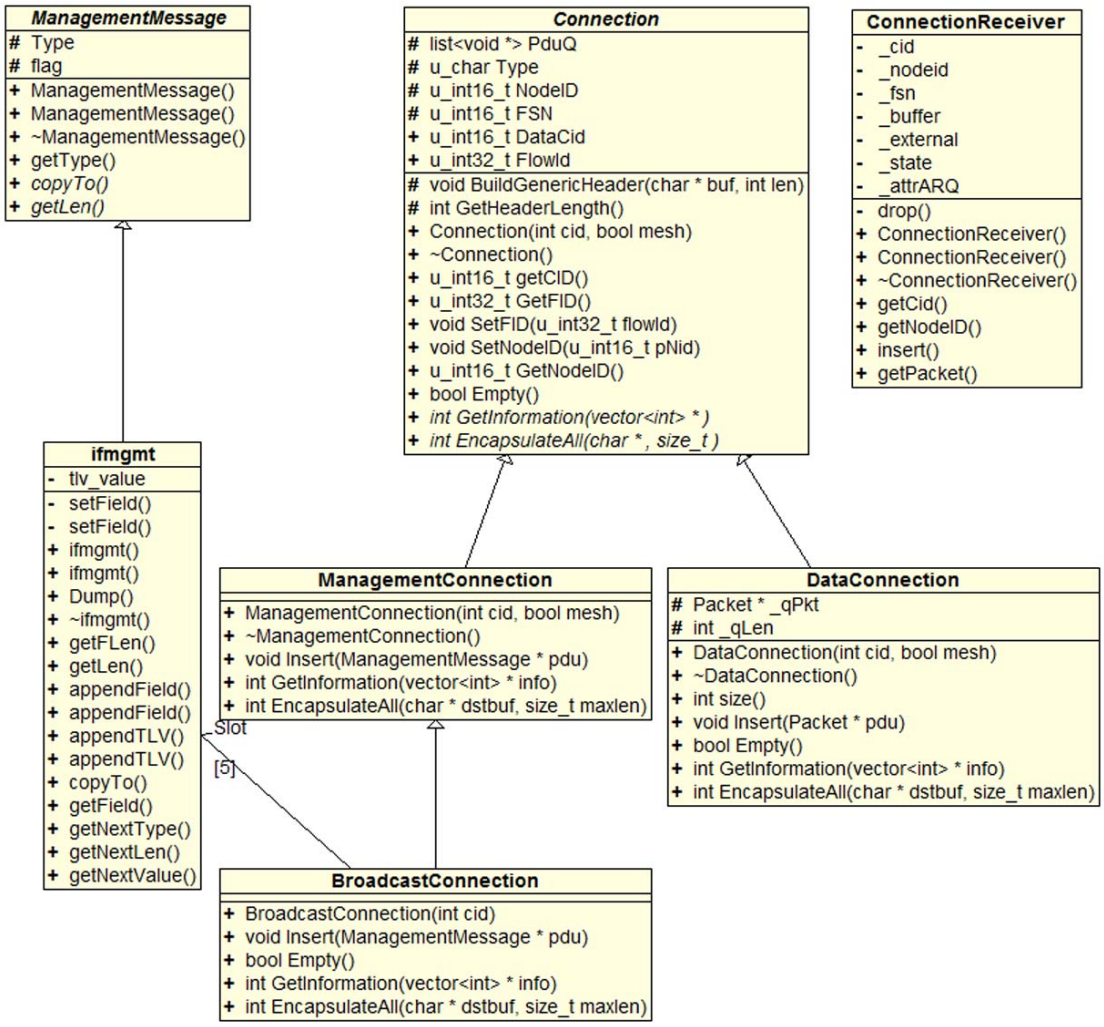

Figure 4-5 Class Diagram of Connection Hierarchy

The *ManagementConnection class* which serves for management connections stores items of *ManagementMessage* type. Since the broadcast connection must transmit broadcast messages in a specific order instead of first-in first-serve behavior, the *BroadcastConnection class* which is a specialized *ManagementConnection* uses message slots to store these broadcast messages in order. The *DataConnection class* which serves for data transport connections stores items of *Packet* type. The *DataConnection class* is the only one that supports the PDU fragmentation in current. The *ConnectionReceiver class* is used to receive and reassemble the payload from

MAC PDU into original format, either management messages or data packets.

To recall that each SS has three management connections and one (or more) data transport connection, each SS Object agent (*ssObject class*) and SS node (*mac802_16_PMPSS class*) both correspondingly have three *ManagementConnection* to maintain the common management connections and a list of *DataConnection* for data transport connection. The BS node only maintains global management connections, such as broadcast connection and initial ranging connection, by itself.

4.2. The Architecture of Physical Module

In this section, we introduce some important classes of the physical layer protocol modules and we also show their class diagram to illustrate the relations between each class.

In physical layer, we separate channel coding and channel model as two independent classes, *ChannelModel* and *ChannelCoding*. Two main portions of physical layer modules are *OFDM_PMPBS* and *OFDM_PMPSS* classes, which are inherited from *OFDM_80216 class*. *OFDM_80216 class* contains *ChannelModel* and *ChannelCoding* to simulate wireless environment and coding scheme. The relation is also shown in Figure 4-6

Common configurations and attributes used in both base station and subscriber station are declared in *OFDM_80216 class*, such as symbol configuration and transmission power. The channel model and channel coding are also two common tasks, and therefore they are declared in *OFDM_80216 class*.

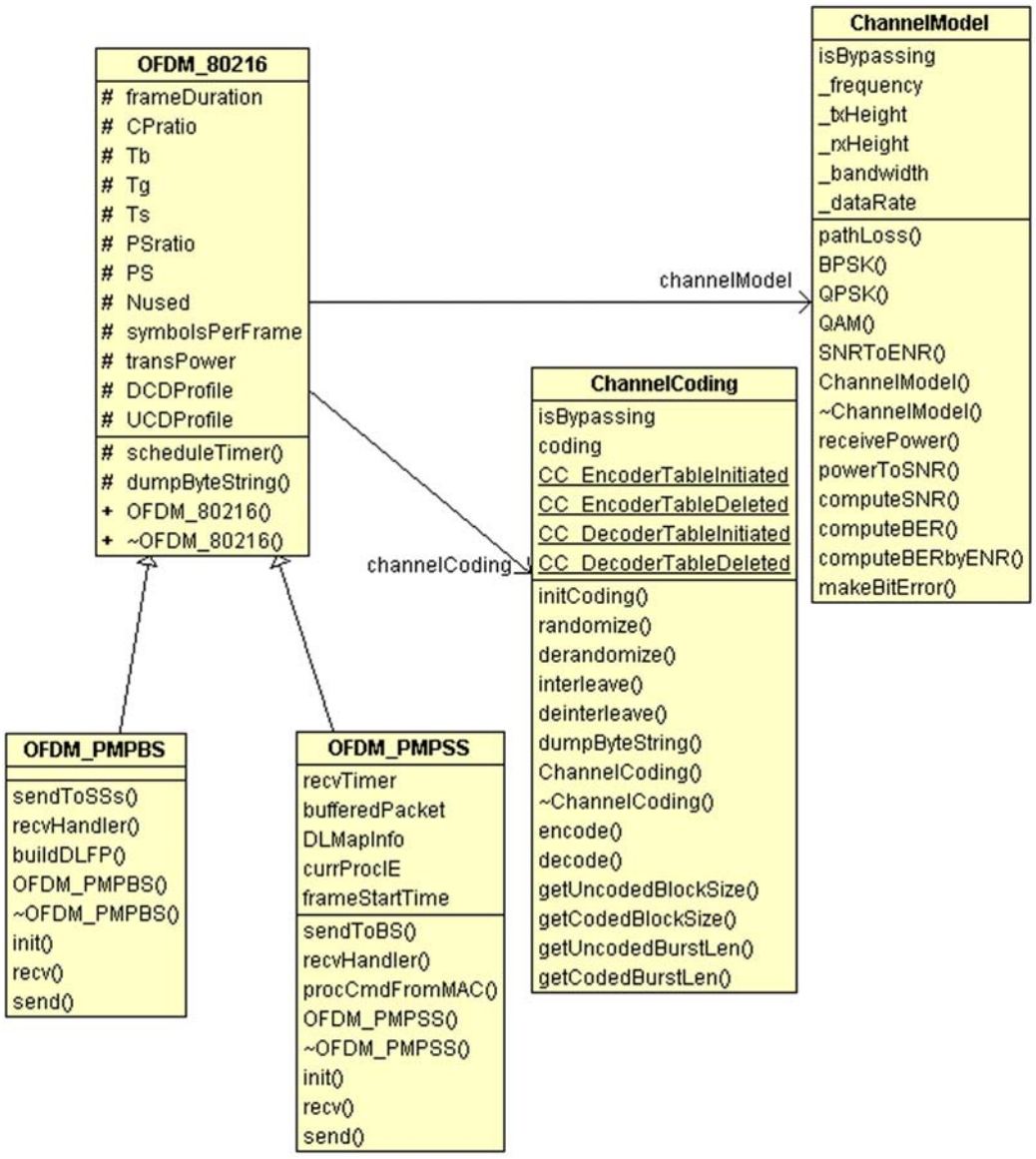

Figure 4-6 Class Diagram of Physical Layer Modules for WiMax PMP-mode

In *ChannelModel class*, we give frequency band, bandwidth, antenna height, and OFDM symbol duration in constructor. The frequency band, bandwidth and antenna height used in calculating path loss is constant since he channel model adopted in our work had contained random variable. The OFDM symbol duration is used to calculate date rate. This may affect error ratio caused by different modulation scheme (BPSK, QPSK, and QAM.)

With given distance, the *pathLoss()* function calculates the magnitude of signal

loss during transmission. Therefore, the *receivePower()* function can compute the received signal power from given transmit power and transmission distance, then the received signal power can be transformed into signal-to-noise ratio (SNR) and energy-to-noise ratio (ENR) by *powerToSNR()* and *SNRToENR()*. The *BPSK()*, *QPSK()*, and *QAM()* functions called by *computeBER()/computeBERbyENR()* calculate uncoded bit error rate with given SNR/ENR under respective modulation. The *makeBitError()* function flips some bits over given raw data stream buffer to simulate bit error and the number of flipped bit is based on the buffer length and uncoded bit error rate.

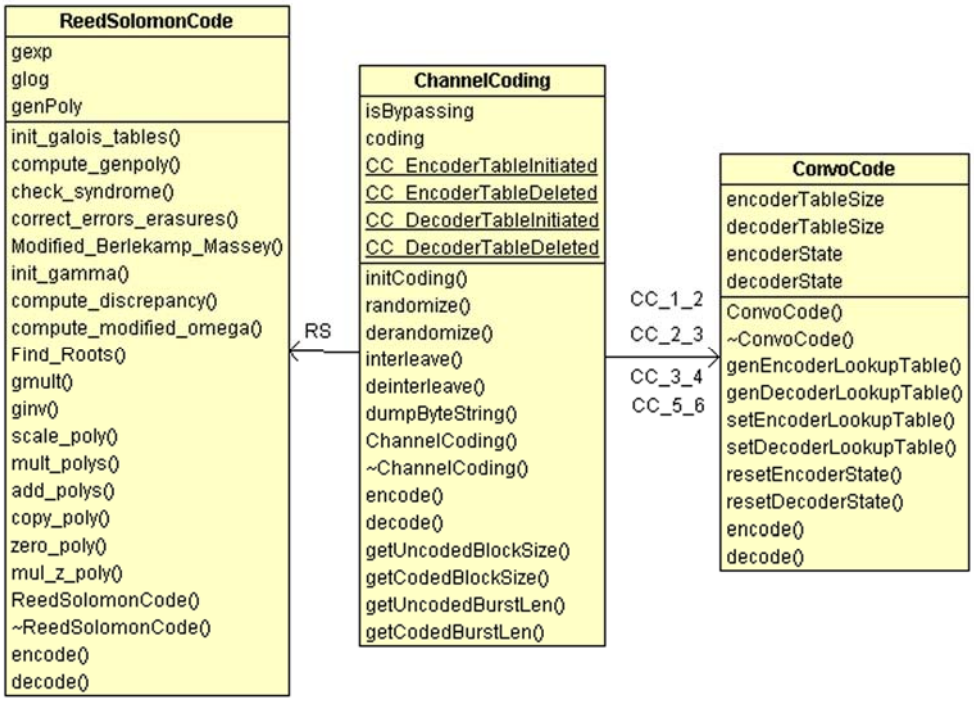

Figure 4-7 ChannelCoding Class Diagram

In *ChannelCoding* class, it provides all steps of coding process, and two important feature Reed-Solomon code (RS) and convolutional code (CC) are split as class respectively. A *ReedSolomonCode class* object and four *ConvoCode class* object is contained in *ChannelCoding class* for supporting multiple coding scheme

simultaneously. Each *ConvolCode class* object takes response to different coding rate since they works according to different tables.

The member function of *ChannelCoding*: *encode()* is called by *OFDM_PMPBS/OFDMPMPSS class* when a data burst is ready to send. It calls *randomize(), ReedSolomonCode::encode(), ConvoCode::encode(),* and *interleave()* function in order. The channel model is then applied after encoding.

ChannelCoding::decode() is called to receive data burst. In reception direction, it proceeds in contrary order with sending direction: *deinterleave(), ConvoCode::decode(), ReedSolomonCode::decode(), derandomize().*

Chapter 5. Simulation Results

In this chapter, we first validate the simulation results in the aspects of latency and throughput. In this chapter, we simulate all cases with the OFDM parameters shown in Table 5-1 [6].

Table 5-1 Basic OFDM parameter

OFDM parameters	Value	Scenario
Bandwidth BW		20 MHz
Sampling rate F_s=1/T	Depends on BW	23.04 MHz
Useful time T_B	256*T	11.11 us
T_G /T_B	1/4, 1/8, 1/16, 1/32	1/4
CP time T_G	T_B +T_G	2.78 us
Symbol Time T_{sym}		13.89 us
Frame Duration	2.5, 4, 5, 8, 10, 12.5, 20	10 ms

5.1. Validation and Analysis of Simulation

In this section, we validate our implementation and also analyze the results of simulations. In experiments shown in this section, the channel error model is disabled to reduce too much variance of the simulated results.

5.1.1. Round-Trip Time (RTT)

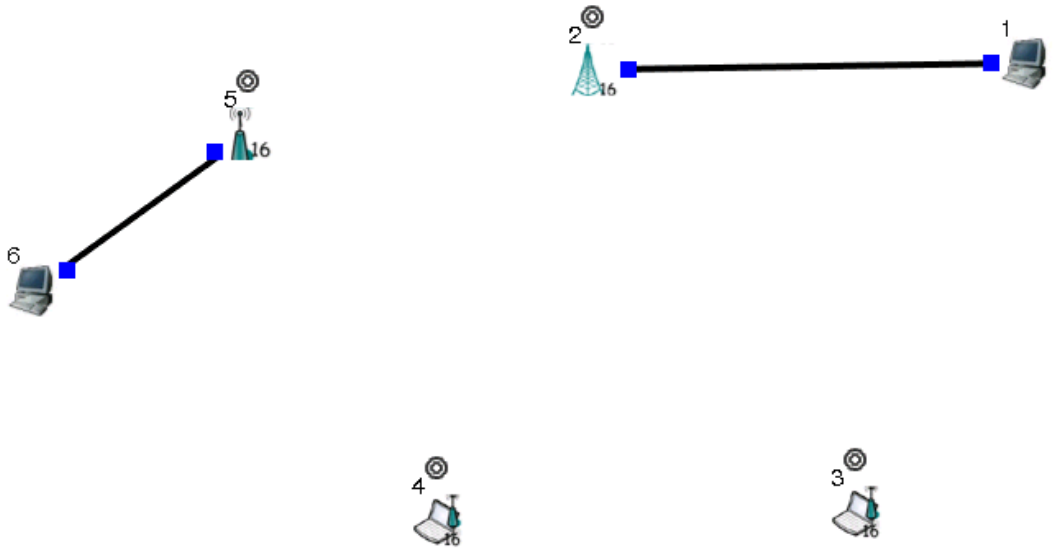

Figure 5-1 Example scenario for RTT validation

Node1: IP 1.0.1.1, Node3: IP 1.0.4.2, Node4: 1.0.4.3, Node6: IP 1.0.2.2

As shown in Figure 5-1, the scenario contains six nodes. This scenario is used for validating round-trip times experienced by network nodes using WiMAX PMP-mode. We use the basic network utility “ping” to validate the RTT from (1) Node 1 to Node 6, (2) Node 6 to Node 1, and (3) Node 3 to Node 4. The frame duration of the WiMAX network is 10 milliseconds. Each SS node is close enough to use 64QAM 3/4 modulation and the channel error model is disabled to avoid packet losses.

We run the ping program every ten seconds and each ping program sends an ICMP echo request per second. The first ping program starts at fifth second and ends after receiving the reply of last request send at thirteenth second. In other words, the number of ICMP echo request packets a ping program sends is totaled as nine. The second ping starts at 15.001 second and ends after 23.001 second. We run each ping program at different time tick to observe to the changes of RTT.

Case 1:

First, we run “ping –c 9 1.0.1.1” to node 1 on node 6. The setting is shown in Figure 5-3 and the result of first two ping is also shown in Figure 5-4. Because the IP packet crosses Node 2 (BS Gateway) and Node 3 (SS Gateway), its TTL (Time-To-Live) value was minus 2 from 64. The first ping start at 5.000 second results RTT of 10.1 ms, and the second ping start at 15.001 second results RTT of 19.1 ms.

The first ICMP request was sent from Node 5 at 50000000th tick (5.000 second) and arrived in Node 6 at 50001923rd tick. There is a frame during 50000000th ~ 50100000th tick and the BS schedule Node 5 to send at 50005144. With consideration of timing advance, Node 5 must send the packet at 50004862nd tick. Therefore the ICMP packet was transmit on this frame. Then Node 2 (BS Gateway) forwards the ICMP packet to Node 1, and receives the ICMP-REPLY packet at 50007840th tick. In next frame (during 50100000th tick ~ 50200000th tick), the ICMP-REPLY packet was sent. Finally, the ICMP-REPLY packet arrives in Node 6 at 50101576th tick. Thus, the measured RTT calculated by 50101576-50000000 was 101576 ticks, 10.1 milliseconds. In the second traffic ping starting at 15.001 second, because the ICMP packet is always too late to get the frame of the time, it must wait more 9 milliseconds than previous ping for next frame to transmit.

```
#nctuns traffic generator file
$node_(6)  5.000000 130.000000 ping -c 9 1.0.1.1
$node_(6) 15.001000 130.000000 ping -c 9 1.0.1.1
$node_(6) 25.002000 130.000000 ping -c 9 1.0.1.1
$node_(6) 35.003000 130.000000 ping -c 9 1.0.1.1
$node_(6) 45.004000 130.000000 ping -c 9 1.0.1.1
$node_(6) 55.005000 130.000000 ping -c 9 1.0.1.1
$node_(6) 65.006000 130.000000 ping -c 9 1.0.1.1
$node_(6) 75.007000 130.000000 ping -c 9 1.0.1.1
$node_(6) 85.008000 130.000000 ping -c 9 1.0.1.1
$node_(6) 95.009000 130.000000 ping -c 9 1.0.1.1
$node_(6) 105.01000 130.000000 ping -c 9 1.0.1.1
```

Figure 5-2 traffic generator file

Figure 5-4 and Figure 5-5 illustrates the moving flowchart of the ICMP packet and explains the result RTT. In Table 5-2, all results of RTT validations are list.

```
64 bytes from 1.0.1.1: icmp_seq=0 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=1 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=2 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=3 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=4 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=5 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=6 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=7 ttl=62 time=10.1 ms
64 bytes from 1.0.1.1: icmp_seq=8 ttl=62 time=10.1 ms
--- 1.0.1.1 ping statistics ---
9 packets transmitted, 9 packets received, 0% packet loss
round-trip min/avg/max = 10.1/10.1/10.1 ms

64 bytes from 1.0.1.1: icmp_seq=0 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=1 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=2 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=3 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=4 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=5 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=6 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=7 ttl=62 time=19.1 ms
64 bytes from 1.0.1.1: icmp_seq=8 ttl=62 time=19.1 ms
--- 1.0.1.1 ping statistics ---
9 packets transmitted, 9 packets received, 0% packet loss
round-trip min/avg/max = 19.1/19.1/19.1 ms
```

Figure 5-3 ping results and statistics

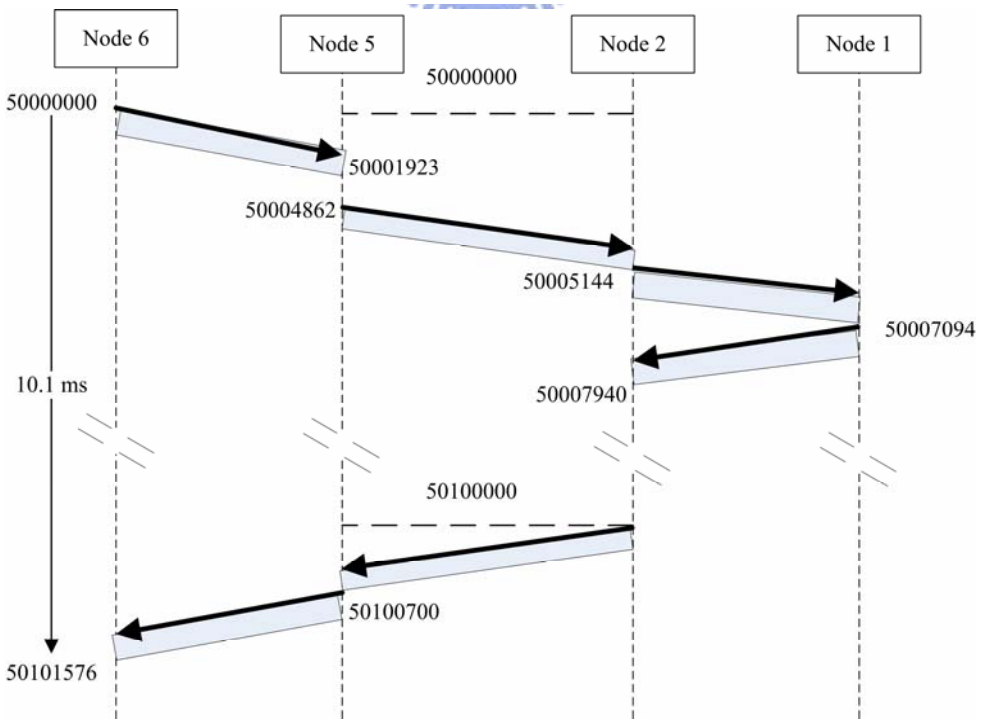

Figure 5-4 packet flow chart from Node6 to Node1

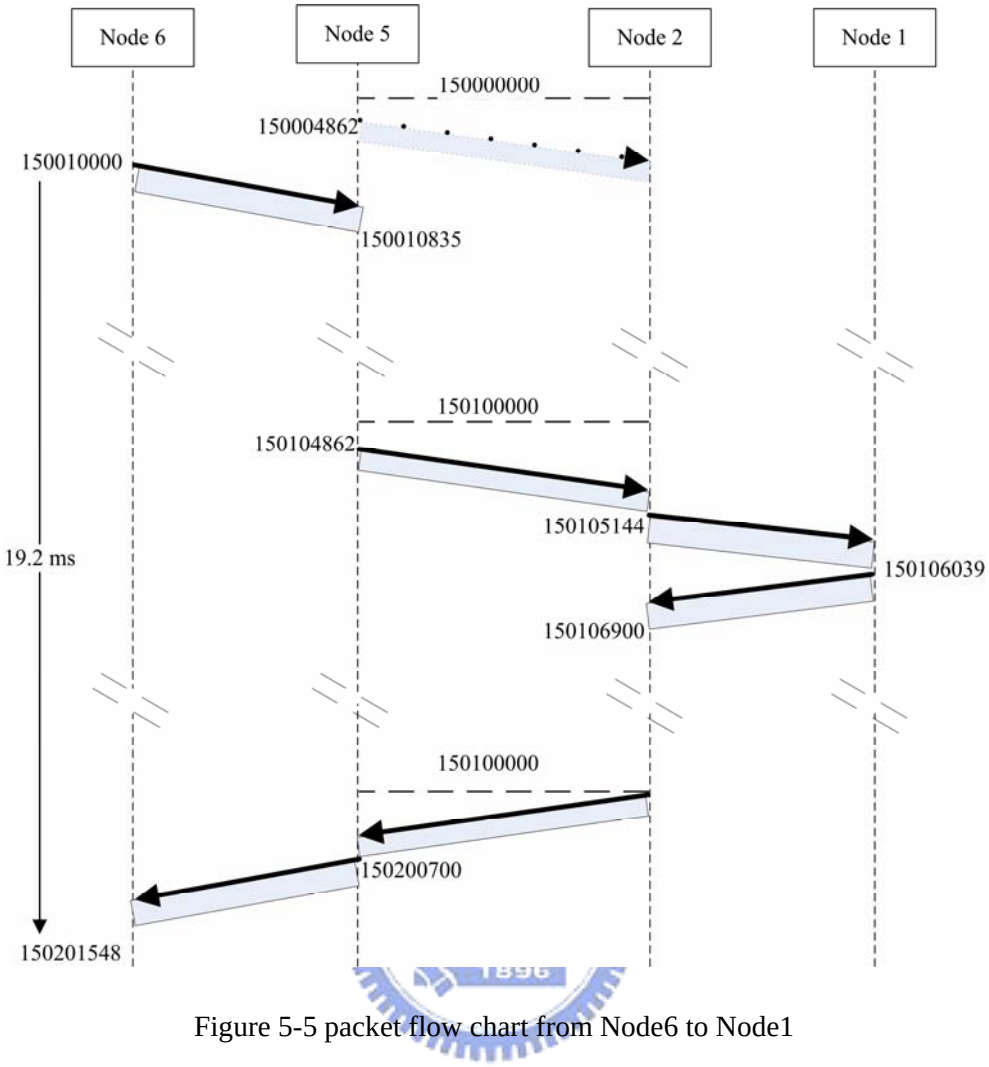

Figure 5-5 packet flow chart from Node6 to Node1

Case 2:

Case 2 is contrary to case 1 on the direction of ICMP traffic. Since the downlink traffic occurs at the beginning of frame, the ICMP request packet always misses the frame of the time. It was waiting in the queue till next frame duration. However, after Node 5 SS Gateway receives and forwards the ICMP request packet to the Node 6, Node 6 replies the ICMP reply packet very soon (2a). Therefore, Node 1 gets the ICMP reply packet in the second frame duration. Otherwise, the ICMP reply packet must wait again till the third frame duration.

To verify the effect of delay between Node 5 and Node 6, we set the propagation delay of the link between Node 5 and Node 6 larger than Node 5's uplink start time. And we repeat this experiment, the results was also list in Table 5-2 (2b).

Table 5-2 the simulation results of ping

Time (sec)	(1) RTT (ms)	(2a) RTT (ms)	(2b) RTT (ms)	(3) RTT (ms)
5.000	10.1	10.6	20.6	20.0
15.010	19.1	9.6	19.6	29.0
25.020	18.1	8.6	18.6	28.0
35.030	17.1	7.6	17.6	27.0
45.040	16.1	6.6	16.6	26.0
55.050	15.1	5.6	15.6	25.0
65.060	14.1	4.6	14.6	24.0
75.070	13.1	3.6	13.6	23.0
85.080	12.1	2.6	12.6	22.0
95.090	11.1	1.6	11.6	21.0
105.100	10.1	10.6	20.6	20.0

Case 3:

In the Case 3, we validate the RTT between two SS node (Node 3 and Node 4). Since only the first ping start at 5.000 second can get the first frame of the time, it has the RTT only 20 millisecond. Others are wait one more frame duration. The packet moving flowchart of case 3 are also shown in Figure 5-6.

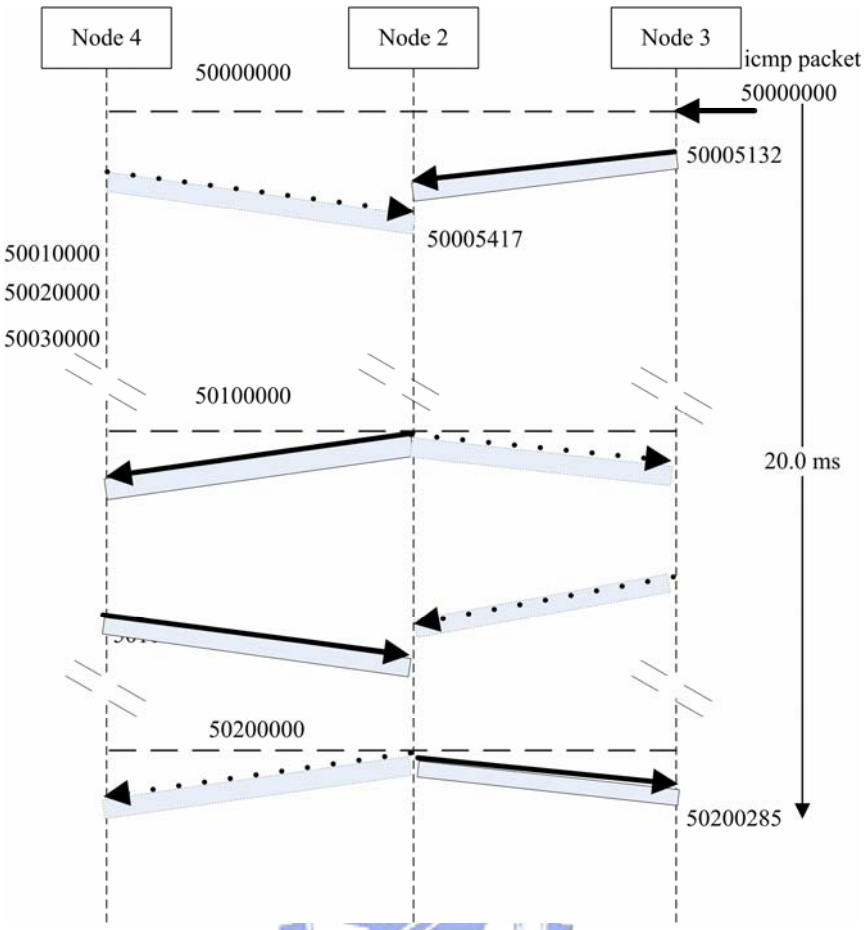

Figure 5-6 packet flow chart (from Node3 to Node4)

5.1.2. Throughput

In order to verify the transmission throughput in our simulation of the WiMAX PMP network, we use a 1400-bytes greedy UDP traffic on downlink to examine the application throughput compared to the scheduling. In order to measure the maximum throughput, we disable the error of channel model to avoid the factors of packet loss.

Figure 5-7 Simulation topology for throughput

Table 5-3 Simulated throughput calculation [Mbps]

fec	Mode	Theoretical throughput	Block Size (B)	Scheduled DL Symbols (S_{DL})	Available throughput
0	BPSK 1/2	6.91	12	674	6.470
1	QPSK 1/2	13.82	24	676	12.978
2	QPSK 3/4	20.74	36	677	19.497
3	16QAM 1/2	27.65	48	677	25.996
4	16QAM 3/4	41.47	72	678	39.052
5	64QAM 2/3	55.30	96	678	52.070
6	64QAM 3/4	62.21	108	678	58.578

Because the BS scheduler needs to reserve some symbol allocations for broadcast messages and for the SS node transmitting management messages, some symbols are occupied. Therefore, the downlink (DL) available symbol was shortened, and also reduces the simulated throughput. The reduced available throughput can be calculated by DL symbols multiplying the date bytes per symbol depended on the mode. The available throughput can be calculated:

$$Throughput = \frac{B * S_{DL}}{Frame\ Duration}$$

For example, the BPSK 1/2 available throughput can be calculated as:

$$\begin{aligned} Throughput &= \frac{12\,(bytes) * 674}{10\,ms} \\ &= 808.8\,Kbytes / ms \\ &= 6.4704\,Mbit / s \end{aligned}$$

Table 5-4 shows the results of UDP throughput.

Table 5-4 Results of simulated throughput [Mbps], UDP: 1400 bytes

fec	Mode	MAC Throughput	UDP Throughput	Utilization (%)
0	BPSK 1/2	6.470	6.294	97.3
1	QPSK 1/2	12.978	12.622	97.3
2	QPSK 3/4	19.497	18.972	97.3
3	16QAM 1/2	25.996	25.300	97.3
4	16QAM 3/4	39.052	38.012	97.3
5	64QAM 2/3	52.070	50.680	97.3
6	64QAM 3/4	58.578	57.019	97.3

In this case, the difference between MAC throughput and UDP throughput are cause by protocol overhead. We can verify the utilization as follow:

$$Utilization = \frac{payload}{header + subheader + crc + ipheader + udpheader + payload}$$

Due to the PDU fragmentation at last block, the utilization will be bound by:

$$0.973 = \frac{1400}{6+1+4+20+8+1400} \le Utilization \le \frac{1400}{6+0+4+20+8+1400} = 0.974$$

Thus, the simulated throughput consists with our expectancy.

Figure 5-8 shows the result of previous simulation with different distance in error-free channel. Because the error of channel model was disabled, the throughput is steady without respect to the distance.

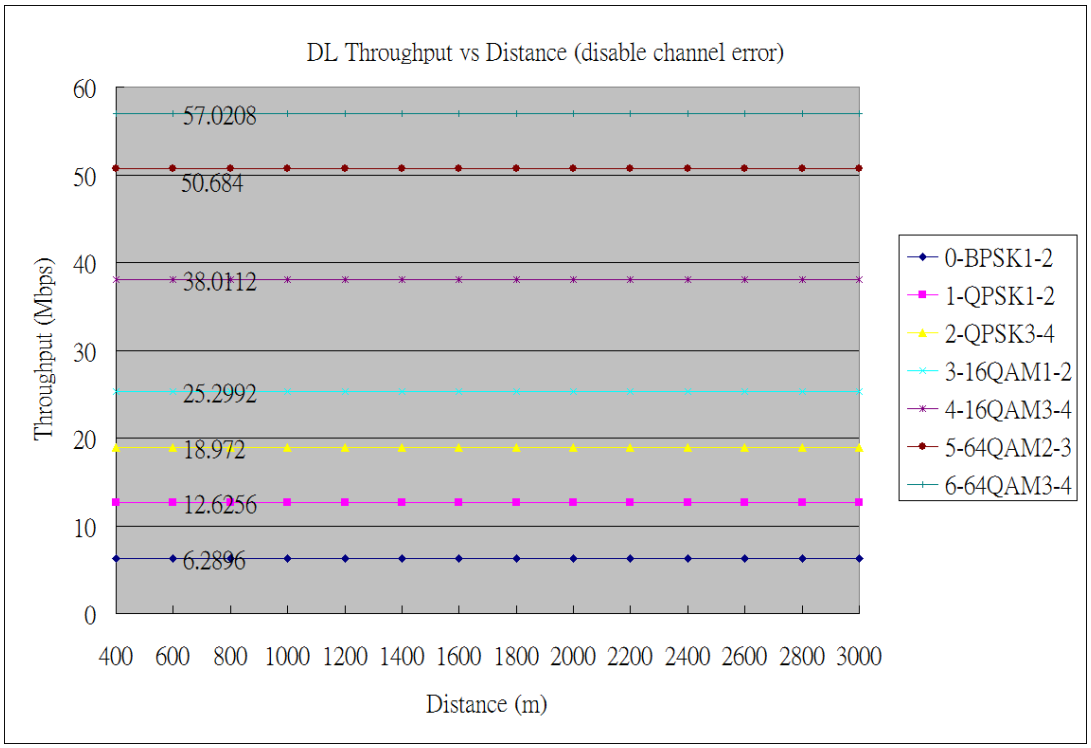

Figure 5-8 UDP throughput vs. Distance (no error)

5.2. Performance Evaluation of WiMax PMP Network

In this section, we will enable the channel error, that is to say the channel error factor will be joined in channel model, to see the performance of WiMAX PMP networks. Unlike what had shown in Figure 5-8 has very perfect performance with error-free channel, we want to see the throughput that reduced by the packet error and packet loss. We use the same scenario shown in Figure 5-7 and the same configurations except for the errors of channel.

$$P_{TX,BS} = 37dBm \qquad Height_{BS} = 80m$$
$$P_{TX,SS} = 23dBm \qquad Height_{SS} = 10m$$

At first part, Figure 5-9 ~ Figure 5-15 illustrate the snapshots of throughputs compared with different distance for each mode. The subscriber station initiates

network entry procedure from 0 second and traffics start at 5th second of simulation time. Due to the packet loss, the network entry procedure may complete latterly. This phenomenon both occurs in 64QAM 2/3 over 1200 meters and 64QAM 3/4 over 110 meters. As shown in figures, throughput is close to the maximum throughput at short range, and reduced by distance. Since the SNR is decreased and varying widely by longer distance (larger path loss), the variance of throughput is become larger.

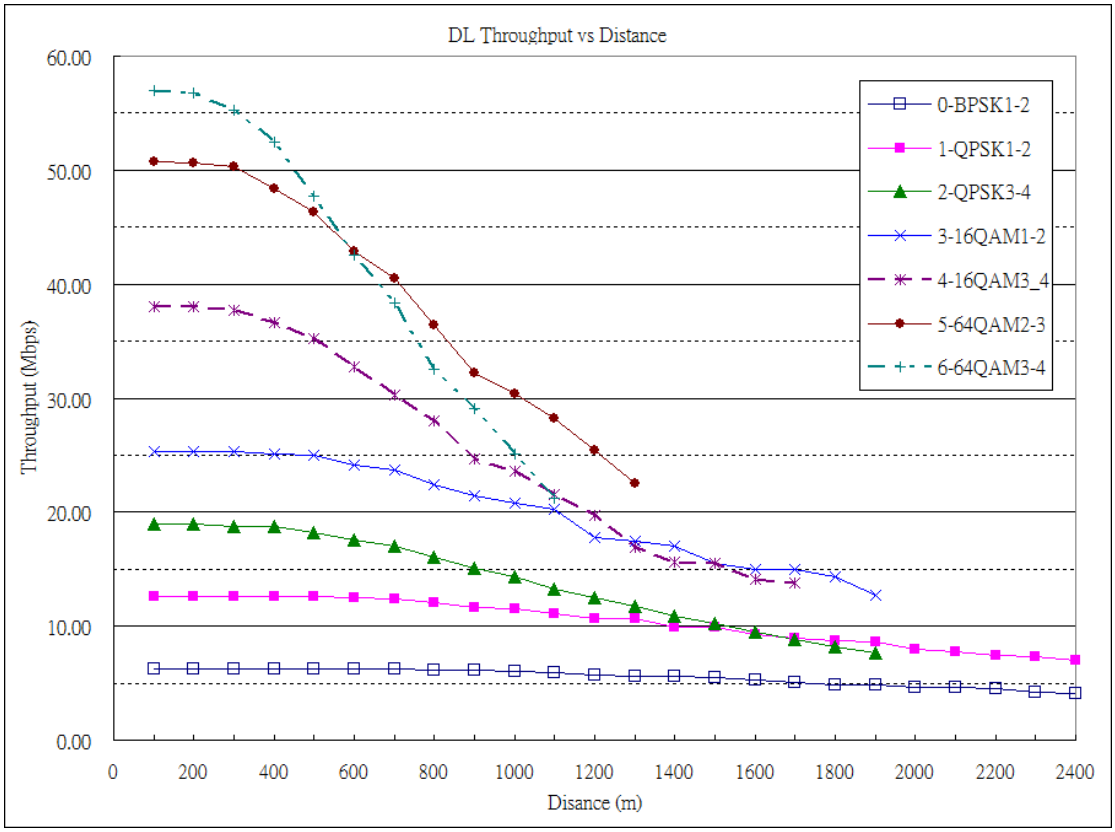

Figure 5-16 gives the overall throughput compared with different distances for each mode. The higher coding rate and modulation scheme gives higher data rate, but shorter coverage range. Higher modulation scheme conveys more data in the same time, but the signal-to-noise ratio is decrease more rapidly. The same modulation scheme has the same bit error probability, but a robust coding scheme corrects more errors at the cost of data rate. As shown in

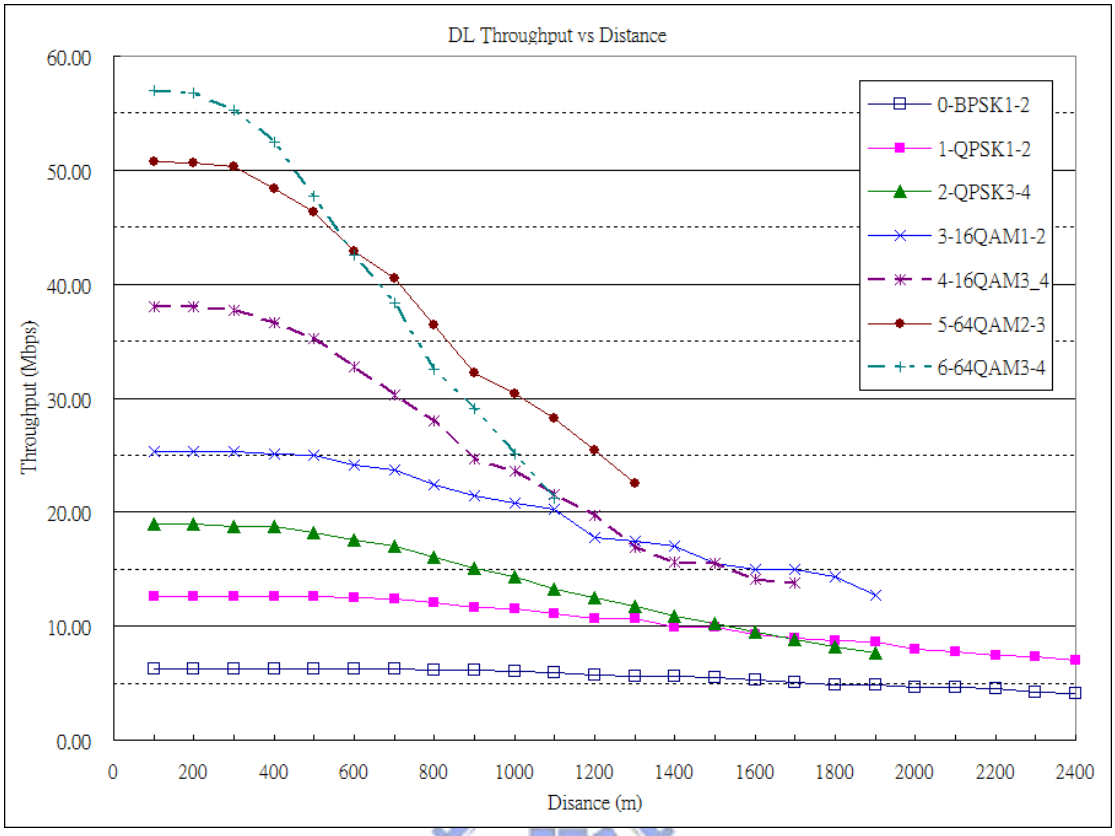

Figure 5-16, QPSK3/4 has lower throughput (8.8 Mbps) than 16QAM1/2 (8.9 Mbps) when the distance is over 1700 meters. 16QAM3/4 has lower throughput (16.9 Mbps) than 16QAM1/2 (17.4 Mbps) when the distance is over 1300 meters. And the throughput of 64QAM3/4 (42.5 Mbps) is lower than which of 64QAM2/3 (42.8 Mbps) over 600 meters.

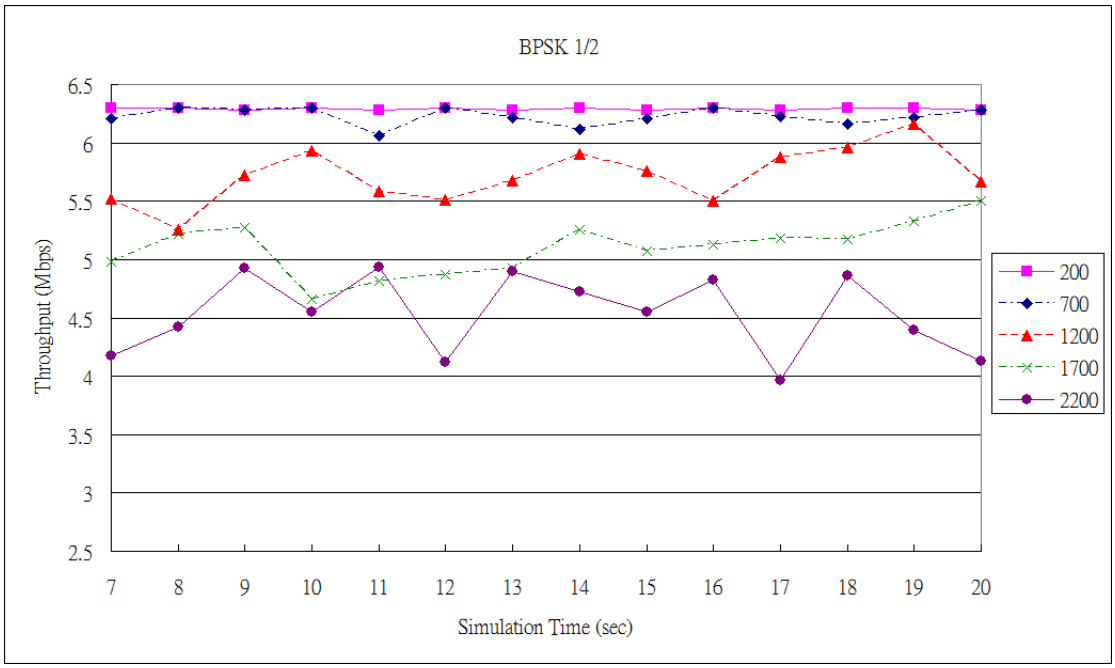

Figure 5-9 BPSK 1/2 UDP Throughput under difference distance

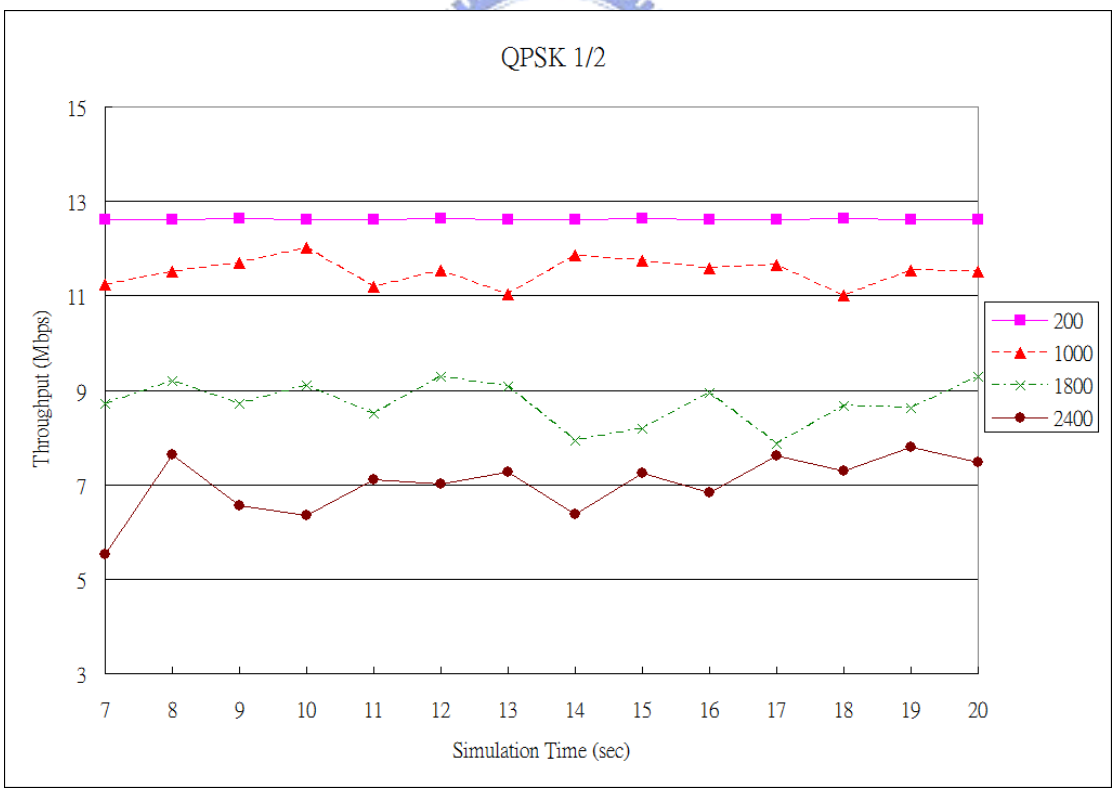

Figure 5-10 QPSK 1/2 UDP Throughput under difference distance

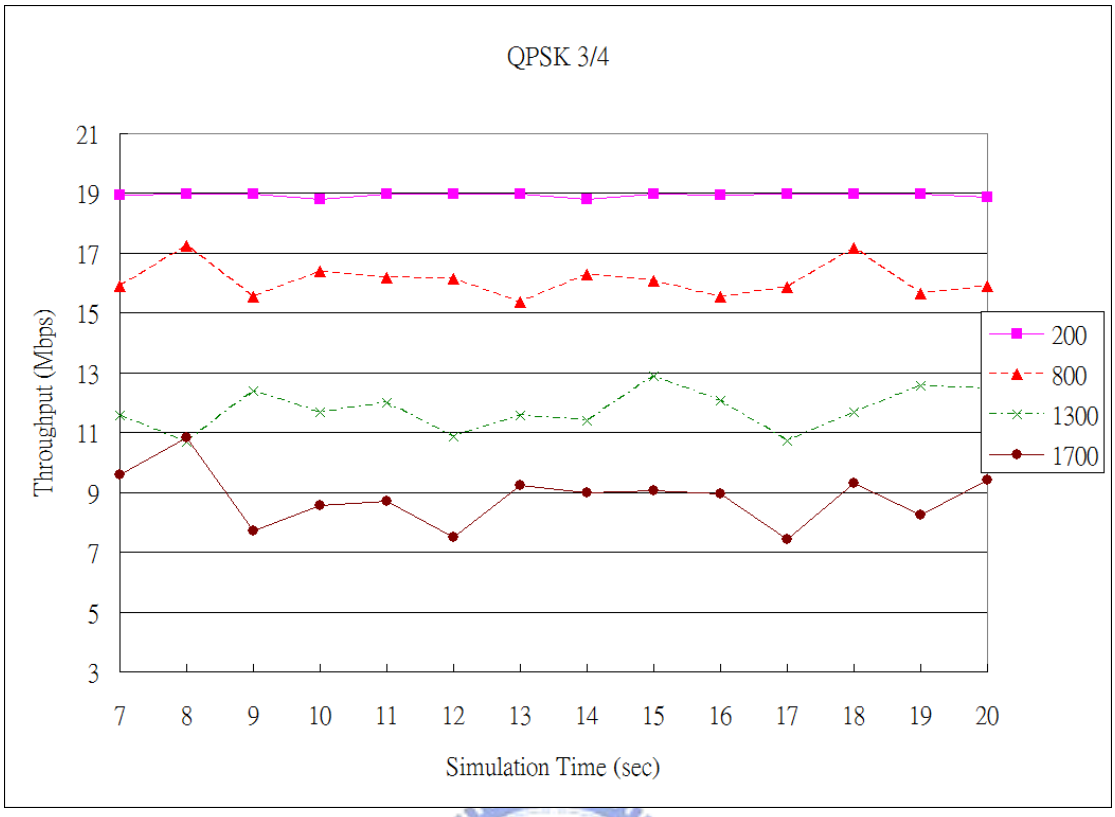

Figure 5-11 QPSK 3/4 UDP Throughput under difference distance

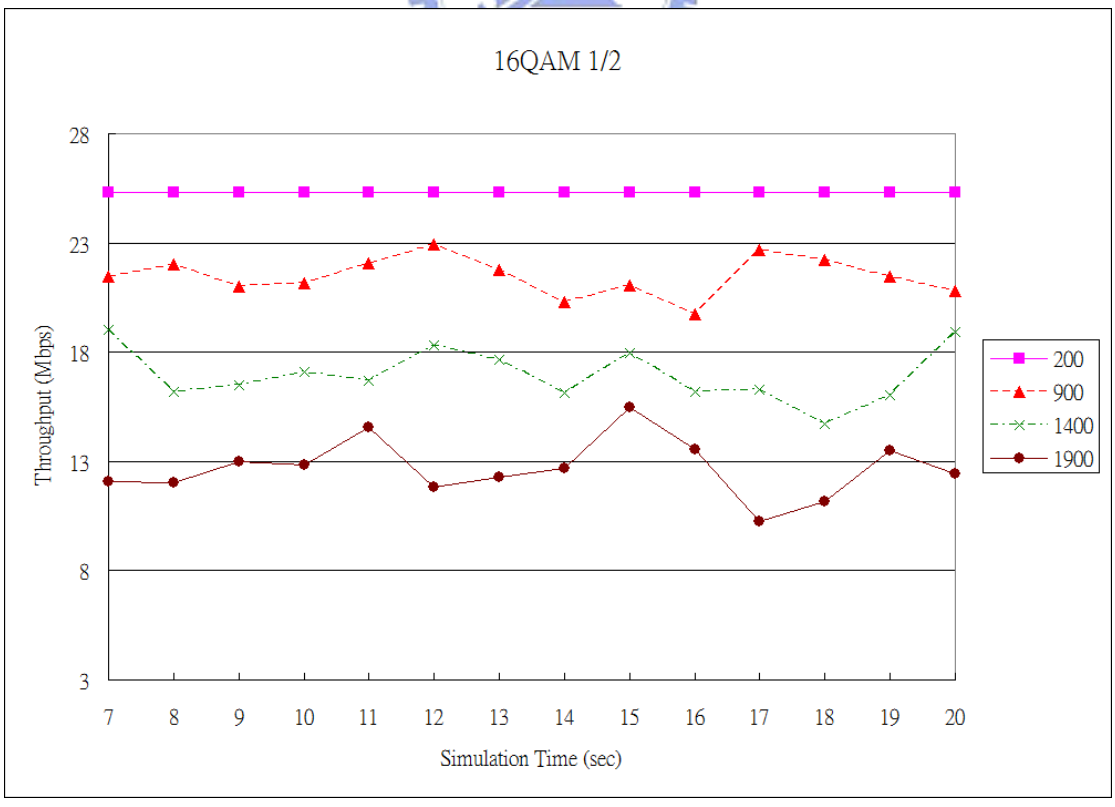

Figure 5-12 16QAM 1/2 UDP Throughput under difference distance

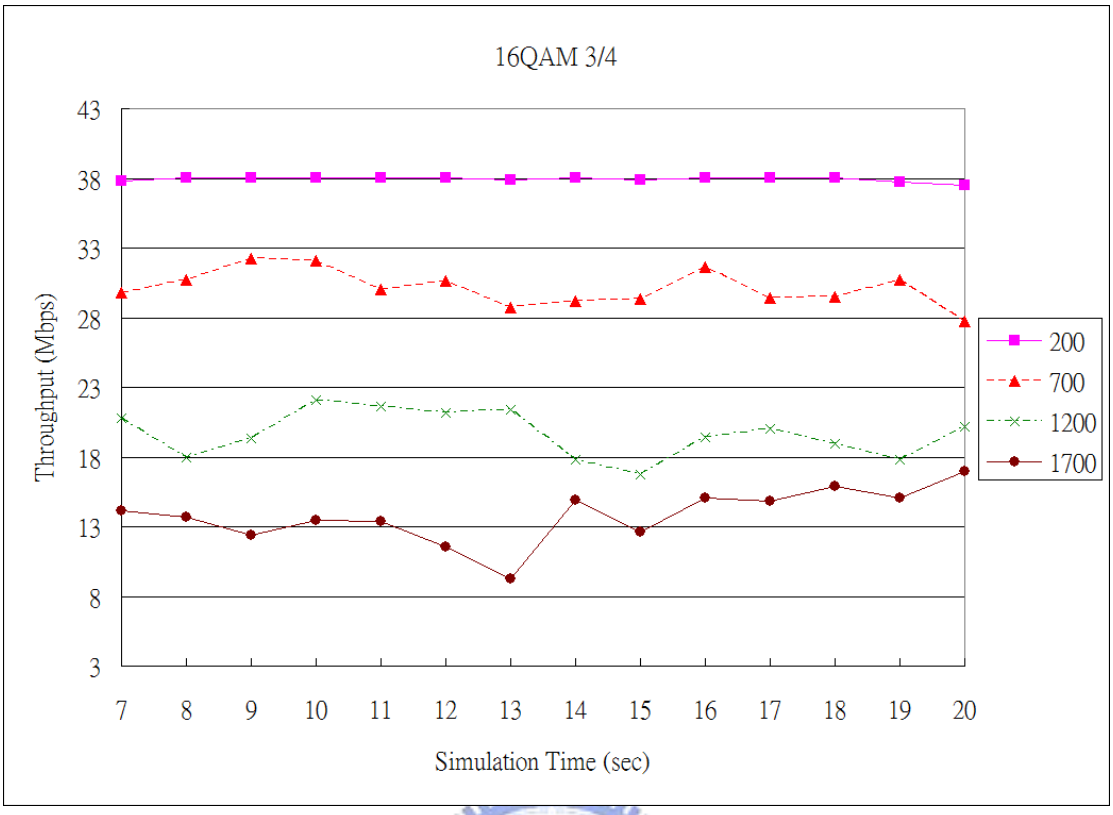

Figure 5-13 16QAM 3/4 UDP Throughput under difference distance

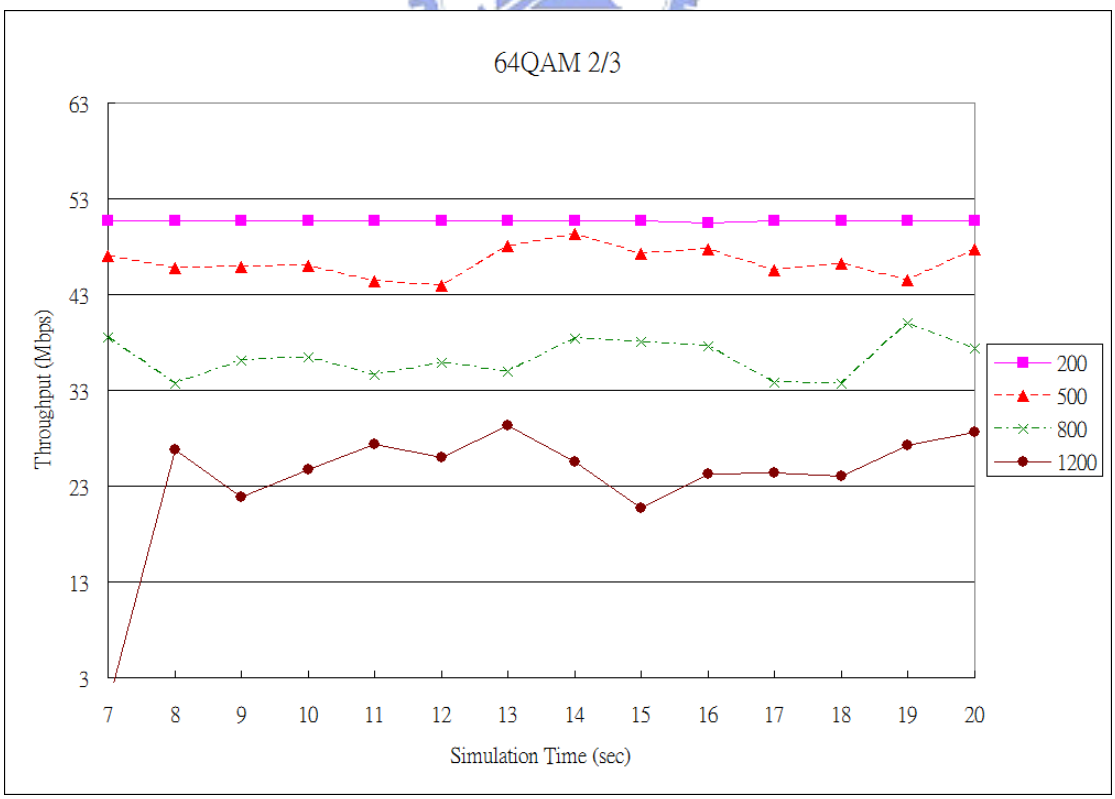

Figure 5-14 64QAM 2/3 UDP Throughput under difference distance

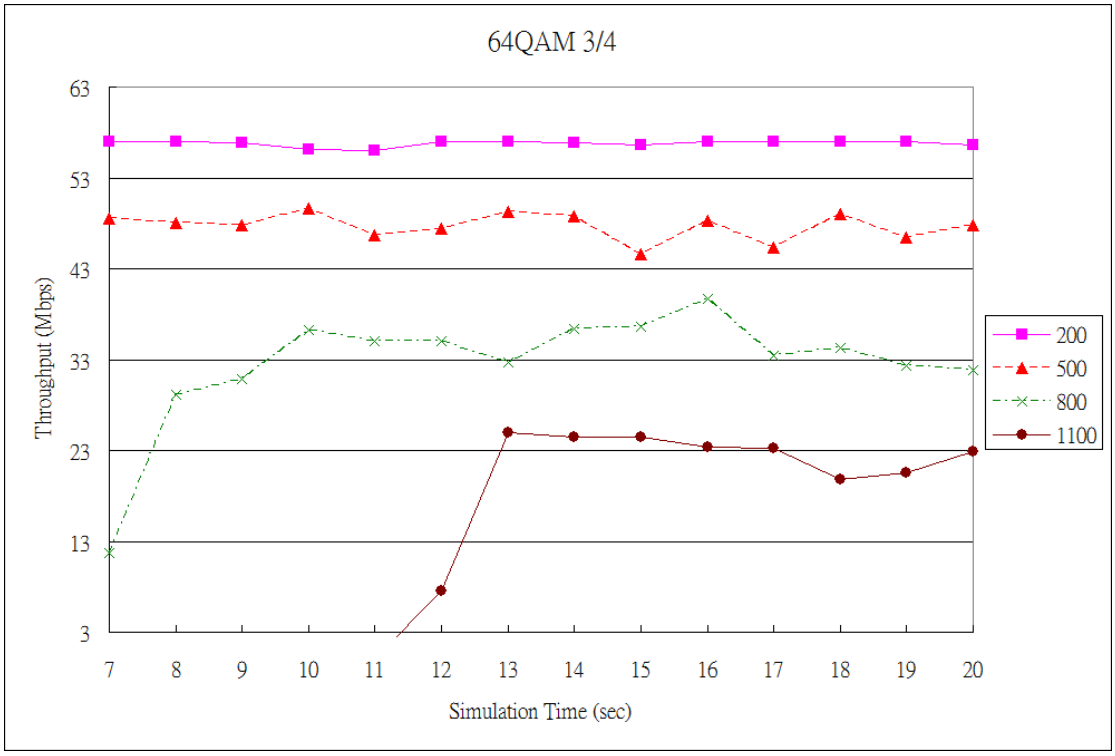

Figure 5-15 64QAM 3/4 UDP Throughput under difference distance

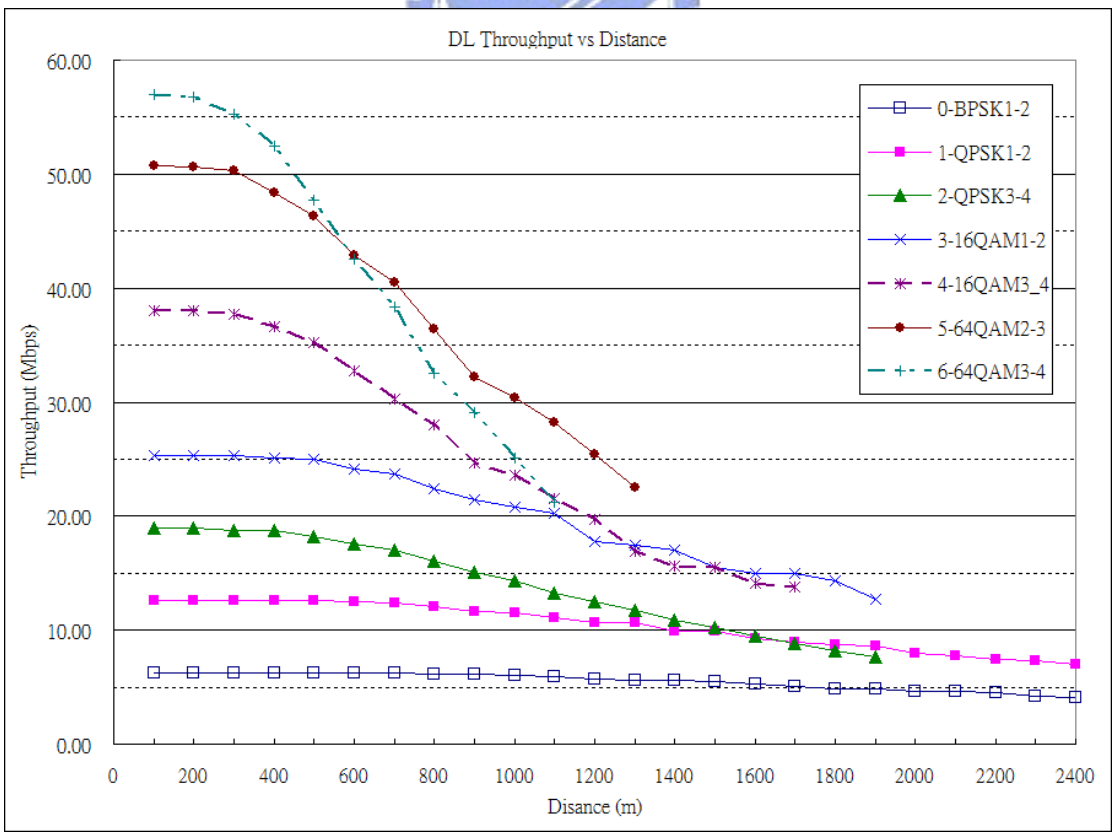

Figure 5-16 Throughput vs. Distance (with error)

5.3. Scalability Analysis

Simulation Machine:

CPU: Pentium4 3.0 GHz

Memory: 2G RAM

OS: Linux, Fedora Core 4 with modify nctuns-2.6.11-kernel

Table 5-5 One BS to one SS with coding, without application traffic

Simulation Time		40sec		80sec		120sec	
fec	Mode	Time (sec)	Mem (KB)	Time	Mem	Time	Mem
0	BPSK 1/2	31	8836	61	8836	91	8836
1	QPSK 1/2	32	8968	64	8968	96	8968
2	QPSK 3/4	26	8968	53	8968	79	8968
3	16QAM 1/2	41	9100	82	9100	122	9100
4	16QAM 3/4	24	9100	47	9100	71	9100
5	64QAM 2/3	32	9232	63	9232	95	9232
6	64QAM 3/4	35	9232	70	9232	105	9232

Table 5-5 shows the minimum simulation execution requirements, time cost and memory usage, since it only contains management messages. The execution time is in proportion to the simulation time and the memory usage is stabled. After analyzing the call graph profile reported by the “gprof” utility, over 60% execution time is taken by convolution decoding. Because the convolution code with code rate 5/6 conveys more data than code rate 2/3, it uses fewer symbols. Therefore, since the size of management messages contained is constant, QPSK 3/4 and 16QAM 3/4 may process faster than QPSK 1/2 and 16QAM 1/2.

Table 5-6 gives the execution time with greedy traffic. It simulates 40 seconds and there are 5 seconds greedy traffic from the BS to the SS. Duo to the higher data rates, the execution time is greatly growing. Almost all computations are bound on the convolution decoding. However, because the convolution decoder computes and

accesses memory bits by bits, it is matter of course to use much CPU time. Seeing that the convolution decoding is the major procedure of simulation, the execution time is depending on the decoding data (traffic) size. In the MAC layer modules, each connection can buffer at most 4000 packets (configurable), this queue length must accommodate the target data rates and frame duration. Each packet contains a 1428 bytes IP packet and other data structure used in the NCTUns network simulator. Therefore the growth of memory usage, 8MB, is reasonable.

Table 5-6 One BS to one SS with 5 second greedy traffic

Simulation Time		40 seconds with 5 seconds greedy traffic	
fec	Mode	Time (sec)	Mem (KB)
0	BPSK 1/2	486	17740
1	QPSK 1/2	1063	18004
2	QPSK 3/4	1345	18048
3	16QAM 1/2	2252	18280
4	16QAM 3/4	2992	18340
5	64QAM 2/3	4117	18536
6	64QAM 3/4	4437	18672

In order to verify the influence of data rate, we are experimenting with different traffic and coding configuration as Table 5-7. The first case same as Table 5-6 is experimental group. The second case disables channel coding and channel errors. In the third case, we enable channel coding and channel errors and add a dummy SS into the WiMAX network but there is no application traffic sent to it. In the final case, we add a greedy UDP traffic to the dummy SS added in previous case.

Because the channel coding and channel error factor was disabled in case 2, it simulates in several seconds. However, the throughput is always high abidingly and lost in accuracy.

Table 5-7 The influence of different configuration

Simulation Time		40 second, (5 second greedy traffic)							
Simulation Configuration		One SS, with coding, 1 greedy traffic		Without coding		Two SS		Two SS, two traffic	
fec	Mode	Time (sec)	Mem (KB)	Time	Mem	Time	Mem	Time	Mem
0	BPSK 1/2	486	17740	4	17700	947	17788	1005	26680
1	QPSK 1/2	1063	18004	5	17820	2087	18020	2123	26848

Although there is no traffic sent to the new added SS in case 3, the dummy SS is still receiving and decoding data. Because double SS are in decoding, double decoding data size and double execution time are resulted.

In case 4, there are two greedy traffics sent to the two SS respectively. Since they share the same medium, the amount of data rate are fixed. There is no much extra data to decode than case 3. Therefore, the execution time is similar to case 3. And within the BS, there are two connections which are full of packets. It consumes another 8MB memory spaces.

According to the analysis as above, the execution time is in proportion to the processed data rate, and the memory usage is depending on the traffics and connection queue length. Since the simulation is CPU bound, we can predict the possible ranges of execution time and memory usage according to the basic cases listed in Table 5-6 and Table 5-7.

Another scalability issue, the supported scale of WiMAX network simulation module, it depends on the NCTUns network simulator. The execution limitations are described as above. Since we use the IP-CS scheme in our design and the NCTUns supports class C network, each BS node can support at most 254 SS nodes.

Chapter 6. Future Work

We illustrated the main ideas of design and implementation of our work. In addition, we also showed and proved the correctness of our work. However, the functionalities included in the WiMAX technology are multitudinous. We do not develop all of the capabilities defined in the standard. In this chapter, we list several interesting tasks and some optional features, which are worthy to do in the future.

- QoS signaling and BW request/grant for QoS service

  The IEEE 802.16 Standard defines four classes of QoS support. Currently, we use a simple approach as displacement for QoS and the functionalities of QoS is needed to supplement.

- Automatic Repeat Request (ARQ) connection

  ARQ, which processing retransmits MAC SDU blocks that have been lost or garbled, can be enabled or disabled on the connection basis. Enabling ARQ on connections may increase or decrease the throughput and latency of WiMAX network. Using ARQ can help to improve reliability of TCP. The effect of ARQ on the WiMAX network maybe can compare with our results.

- IEEE 802.16e and OFDMA

  IEEE Standard 802.16e is an advanced version of 802.16. It supports both fixed and mobility, and reinforces some other features such as hold off and power saving for support mobility. The 802.16e works on the OFDMA specification which provides more flexible and more complex allocation. Although there are many differences between 802.16d and 802.16e, we still provide a reference

implementation for it. Based on this reference implementation, we can extend these modules to support 802.16e.

- Mesh/Relay

  Another possible extension of our work is the Mesh/Relay project, or called 802.16j Mobile Multi-hop Relay (MMR). It is expected to extend coverage and enhance throughput to the non-line of sight area. Since it uses relay station like as Mesh mode, it is compatible with PMP mode. Due to the inconsistency between PMP and Mesh mode, supporting this relay functionality is in demand.

Chapter 7. Conclusion

Simulation is a convenient approach to analyze and evaluate networks. It is more convenient than experiment using real machines and more accurate than mathematical modeling. We developed a WiMAX 802.16 network simulation system based on the famous NCTUns network simulator. In our design, the simulated network supports base station nodes and two kinds of subscriber station nodes.

In previous chapters, we explained what we did and how we did to support simulating 802.16 in NCTUns, including the architecture of protocol modules, the designs of the MAC and physical layers. The MAC layer focuses on the management control and the integration with NCTUns. The main functions of the physical layer are performing channel coding and simulating channel model. The channel coding in our implementation performs real bit-level encoding/decoding operations to provide the highest fidelity. For channel modeling, the Rayleigh fading and an empirically based path loss model is adopted.

Furthermore, the simulation results in our implementation are presented, validated, and analyzed. We also discuss the scalability issue. In our implementation, simulations are conducted with encoding/decoding procedures operating at coding block level to obtain more realistic results at the cost of slowing down the simulation speed significantly.

In conclusion, we developed a useful WiMAX PMP-mode network simulator over NCTUns. Based on our work network researchers can develop their protocol modules over our platform, obtaining more realistic results, and analyzing their results in a more efficient way.

Reference

[1] Vinko Erceg, et al., “An Empirically Based Path Loss Model for Wireless Channels in Suburban Environments,” IEEE Journal on selected areas in communications, Vol.17, No.7, July 1999.

[2] Vinko Erceg, K.S. Hari, M.Smith, D. Baum, K. Sheikh, C. Tappenden, J. Costa, C. Bushue, A. Sarajedini, R. Schwartz, D. Brandlund, T. Kaitz, and D.Trinkwon. “Channel Models for Fixed Wireless Applications.” July 2001.

[3] Theodore S. Rappaport, "Wireless Communications - Principles & Practice," Prentice Hall, 1996, p.286.

[4] Marvin K. Simon, Mohamed-Slim Alouini, "Digital Communication over Fading Channels - A Unified Approach to Performance Analysis," John Wiley & Sons, 2000, p.223. [2nd Editionp.256 (8.112)]

[5] RSCODE project, http://rscode.sourceforge.net/

[6] C. Hoymann, “Analysis and Performance Evaluation of the OFDM-based Metropolitan Area Network IEEE 802.16,” Comp. Net., vol. 49, no. 3, Oct. 2005, pp. 341–63.