

SoftMAC in Heterogeneous Wireless Network

By

Jinsong Li

A thesis

presented to the University of Waterloo

in fulfilment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2008

© Jinsong Li 2008

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Wireless networks are growing exponentially by the steady improvement of its speed and quality. IEEE 802.11-based Wireless Local Area Networking (WLAN) has been developed for mobile computing devices in LANs, in a short and limited range. IEEE 802.16 Wireless Metropolitan Area Network (WMAN) is designed for a line-of-sight (LOS) distance with QoS capability. The IEEE 802.11 standard has a totally different MAC layer compared to the IEEE 802.16 standard, normally they will communicate at the Network Layer by switches or routers.

This thesis investigates the major design requirements for SoftMAC design, and will demonstrate a prototype that can meet the design requirements. It proves the possibility and flexibility of using SoftMAC to connect and control Heterogeneous Wireless Network, in order to fulfill seamless handover among multiple heterogeneous wireless interfaces. We will show that by adding the proposed SoftMAC on top of the traditional MAC layer, the mobile station cannot only perform handover between access points, but also essentially open a door to a wider range of application and services.

Acknowledgements

I would like to thank Professor Pin-han Ho for giving me the opportunity to carry out this research project. His encouragement and advice during the entire process is appreciated. I would like to thank Xiaodong Li, James Ho for their hard working and helping in the project. It is a pleasure of working with the team and I am grateful to be a master student of Electrical and Computer Engineering department at the University of Waterloo. My sincere thanks to all those generous individuals who are working on open source projects and kindly replied to my queries. Finally, I would like to thank my family for their constant support and encouragement.

Table of Contents

Abstract.....	ii
Author's Declaration	iii
Acknowledgements	iv
1. Introduction	1
1.1. IEEE 802.11 wireless network	1
1.2. IEEE 802.16 wireless network	3
1.3. SoftMAC in Heterogeneous Wireless Network	4
2. Background	6
2.1. TCP/IP Model	6
2.2. IEEE 802.11 MAC Layer	8
2.3. IEEE 802.16 MAC Layer	11
2.4. Linux Kernel Network Driver	14
2.5. MADWiFi and SoftMAC	16
3. Theory Approaches	19
4. Platform and Implementation	24
4.1. IXP425 Test Board	25
4.2. SoftMAC for MADWiFi Driver	27
4.3. SoftMAC Layer Structure Design	30
4.4. Handover Algorithm	33
4.5. Experimental Results	35
5. Discussion and Future Work	39
References	40

List of Figures

Figure 1.1: Networking Architecture of SoftMAC	5
Figure 2.1: OSI Model	6
Figure 2.2: TCP/IP Model	7
Figure 2.3: IEEE 802.11 Packet Format	10
Figure 2.4: IEEE 802.16 Reference Model and Protocol Stack	12
Figure 2.5: Linux Network Architecture	15
Figure 2.6: Architecture of Atheros NIC driver	16
Figure 3.1: Layer 2.5 SoftMAC architecture	20
Figure 3.2: SoftMAC in Linux Network Architecture	22
Figure 4.1 SoftMAC with 802.11 and 802.16	24
Figure 4.2 IXP425 test board	25
Figure 4.3: Algorithm for the Handover Algorithm	34
Figure 4.4: Seamless Handover Test-Bed	35
Figure 4.5: ifconfig of the Wireless Cards in the Client Station	36
Figure 4.6: Handover Result with SoftMAC	37
Figure 4.7: Handover without SoftMAC	38

List of Tables

Table 1: MADWiFi Required for Linux Kernel compile	27
Table 2: Procedure for MADWiFi Driver Configuration.	28
Table 3: SoftMAC for MADWiFi.	29
Table 4: Structure of SOFTMAC_LAYER_INFO	30
Table 5: Scripts to load SoftMAC	30
Table 6: Procedure for MultiMAC Configuration	31
Table 7: Required Parameters for Intelligent Handover Algorithm	32

Glossary

802.11	IEEE 802.11 standard, Wireless Local Area Network in the 5 GHz and 2.4 GHz public spectrum bands. [1]
WLAN	Wireless Local Area Network, also called “Wi-Fi” (Wireless Fidelity).
802.16	IEEE 802.16 standard, Wireless Metropolitan Area Networks with a line-of-sight (LOS) capability. [1]
WMAN	Wireless Metropolitan Area Network, also called “WiMAX” (Worldwide Interoperability for Microwave Access).
LOS	Line-of-sight capability.
CSMA/CD	Carrier Sense Multiple Access with Collision Detection, CSMA/CD is a modification of pure Carrier Sense Multiple Access (CSMA). Collision Detection is used to improve CSMA performance by terminating transmission as soon as a collision is detected, and reducing the probability of a second collision on retry. [1]
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance, CSMA/CA is a modification of pure Carrier Sense Multiple Access (CSMA). Collision Avoidance is used to improve CSMA performance by checking if the channel is sensed busy before transmission then the transmission is deferred for a "random" interval. This reduces the probability of collisions on the channel. [1]
MAC	Media Access Control, a layer of OSI model.
LLC	Logical Link Control, the upper sublayer of the data link layer. The LLC sublayer is primarily concerned with multiplexing protocols transmitted over the MAC layer (when transmitting) and demultiplexing them (when receiving). [1]
PHY	Physical layer, a layer of OSI model.
SoftMAC	An additional MAC layer original developed by University of Colorado to provide a flexible environment for experimenting with MAC protocols.
QoS	Quality of Service, which is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow. [1]

QAM	Quadrature amplitude modulation (QAM) is a modulation scheme which conveys data by changing (modulating) the amplitude of two carrier waves. [1]
BPSK	Binary phase-shift keying, it is a digital modulation scheme that conveys data by changing, or modulating, the phase of a reference signal (the carrier wave) by using two phases which are separated by 180° and so can also be termed 2-PSK.
NIC	Network Interface Card, a piece of computer hardware designed to allow computers to communicate over a computer network. It is both an OSI layer 1 (physical layer) and layer 2 (data link layer) device. [1]
SNR	Signal to Noise Ratio, is an electrical engineering concept, also used in other fields (such as scientific measurements, biological cell signalling and oral lore), defined as the ratio of a signal power to the noise power corrupting the signal. [1]
WEP	Wired Equivalent Privacy security algorithm, is part of the 802.11a, 802.11b, and 802.11g standards.
MADWiFi	Open source project, Multiband Atheros Linux kernel Driver for Wireless LAN, Atheros chipsets support IEEE 802.11a/b/g.

1. Introduction

With the steady improvement of the speed and quality recent years, wireless networks are growing exponentially. IEEE 802.11 and 802.16 wireless networks are two popular wireless networks, which are designed for different ranges. 802.11 WLAN (WiFi) is developed for mobile computing devices in LANs, in a short and limited range. 802.16 WMAN (WiMAX) is developed to delivery the last mile wireless broadband access as an alternative to cable and DSL. There are many technical differences between the two wireless standards.

1.1 IEEE 802.11 wireless network

IEEE 802.11-based wireless network is called Wireless Local Area Networking (WLAN) or Wireless Fidelity (Wi-Fi). It is deployed very popularly at home, school or company and wide available hardware devices make it the first choice for many users. Wi-Fi is widely available in more than 250,000 public hotspots and tens of millions of homes and university campuses all over the world. Compared to larger range wireless solution, there are many advantages of a 802.11 network, such as lower cost, lower power consumption, inter-operablability with different competitive brands and higher availability.

At the MAC layer, the 802.11 families use CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) medium sharing mechanism instead of the classic Ethernet CSMA/CD (Collision Detection). It is impossible to detect a radio collision in a wireless environment. The 802.11 standard defines the standard protocol and interconnection method of data communication via the air, radio or infrared, in a local area network (LAN). The basic access method for 802.11 is called Distributed Coordination Function (DCF) and it's mandatory for all stations. Another media access control method, the Point Coordination Function (PCF), is an optional extension to DCF. PCF provides a time division duplexing capability to allow the access point to deal with time-bounded, connection-oriented services. Using this method, one AP controls the access through a polling system. CSMA/CA needs each station to listen to other users. If the channel is idle the station is allowed to transmit. If it is busy, each station waits a random back off

period until the other transmission stops. Packet reception in DCF requires acknowledgements (ACK). The period between completion of packet transmission and start of the ACK frame is one Short Inter Frame Space (SIFS). ACK frames have a higher priority than other traffic. Fast acknowledgement is one of the features of the 802.11 standard, because it requires ACKs to be handled at the MAC sublayer.

CRC checksum and packet fragmentation are two other robust features of the 802.11 MAC layer. Each packet has a CRC attached to ensure its correctness. This is different from Ethernet, where higher-level protocols such as TCP handle error checking. The MAC layer is responsible for reassembling the received fragments; this makes the process transparent to higher-level protocols.

Research has discovered significant problems with the 802.11 MAC layer. These problems are:

- 1). Nodes in an 802.11 network with heterogeneous transmission rate will lead to a low throughput.[20]
- 2). Resource allocation policy is not suitable for multi-hop networks.
- 3). It cannot meet QoS requirement for delay-sensitive applications.

A typical IEEE 802.11 Wi-Fi home router using 802.11b or 802.11g with a stock antenna might have a range of 32 m (120 ft) indoors and 95 m (300 ft) outdoors, and the signal are easily obstructed. Wi-Fi performance also decreases exponentially as the range increases. The Wi-Fi pollution, or an excessive number of access points in the area, especially on the same or neighboring channel, can prevent access and interfere with the use of other access points by others. This is caused by overlapping channels in the 802.11g/b spectrum, as well as with decreased signal-to-noise ratio (SNR) between access points.

1.2 IEEE 802.16 wireless network

The IEEE 802.16 family is officially called Wireless MAN with a line-of-sight (LOS) capability. It aims to provide high-throughput wireless broadband connections over a longer distance, to offer a high speed/capacity, low cost, and a scalable solution to extend fiber optic backbones. After the famous "Worldwide Interoperability for Microwave Access" forum to improve its interoperability, 802.16 standard has been called "WiMAX" also.

The first IEEE 802.16 standard, published in April 2002 to enable the delivery of the last mile wireless broadband access to homes, small businesses, and commercial buildings as an alternative to traditional wired connections, such as cable or DSL. IEEE 802.16 supports point-to-multipoint architecture in the 10-66 GHz range, transmitting at data rates up to 120Mbps.

The IEEE 802.16 medium access control (MAC) layer supports many different physical layer specifications, both licensed and unlicensed. Through the 802.16 MAC, every base station dynamically distributes uplink and downlink bandwidth to subscriber stations using Time-Division Multiple Access (TDMA). This is a dramatic difference from the 802.11 MAC.

The 802.16e implemented a number of enhancements, which including better support for Quality of Service and the use of Scalable Orthogonal Frequency Division Multiplexing Access (OFDMA). 802.16e uses Scalable OFDMA to carry data, supporting channel bandwidths of between 1.25 MHz and 20 MHz, with up to 2048 sub-carriers. It supports adaptive modulation and coding, so that in conditions of good signal, a highly efficient 64 QAM coding scheme is used, whereas where the signal is poorer, a more robust Binary phase-shift keying (BPSK) coding mechanism is used. In intermediate conditions, 16 QAM and QPSK can also be employed. Other PHY features include support for Multiple-in Multiple-out (MIMO) antennas in order to provide good NLOS (Non-line-of-sight) characteristics (or higher bandwidth) and Hybrid automatic repeat request (HARQ) for good error correction performance.

The 802.16 MAC describes a number of Convergence Sublayers, which describe how wireline technologies such as Ethernet, ATM and IP are encapsulated on the wireless

interface and how data is classified, etc. It also describes how secure communications are delivered, by using secure key exchange during authentication, and encryption using AES or DES (as the encryption mechanism) during data transfer. Further features of the MAC layer include power saving mechanisms (using Sleep Mode and Idle Mode) and handover mechanisms.

A key feature of 802.16 is that it is a connection-oriented technology. The subscriber station (SS) cannot transmit data until it has been allocated a channel by the Base Station (BS). This allows 802.16e to provide strong support for Quality of Service (QoS).

1.3 SoftMAC in Heterogeneous Wireless Network

SoftMAC is an additional MAC layer originally developed at the University of Colorado to provide a flexible environment for experimenting the MAC protocols. SoftMAC uses a commodity 802.11a/b/g networking card with a chipset manufactured by the Atheros Corporation. Atheros chipset has a software controlled radio with a predefined physical layer but a flexible MAC layer to provide a flexibility format of the transmitted packets, though this flexibility is not generally exposed by network drivers. By the reverse-engineering of the original driver from Atheros, under Linux platform MADWiFi (Multiband Atheros Driver for WiFi) provides a driver that allows extensive control for Atheros chipset over the MAC layer while still allowing the flexible underlying physical layer to define the waveforms. SOFTMAC is developed depends on the MADWiFi, and also includes a software control system that allows its users to address many of the "systems level" issues facing researchers. [9]

To extend the ability of the SoftMAC to control different chipsets and different wireless networks, a standard SoftMAC interface is defined and a SoftMAC control layer is abstracted from the original SoftMAC. Under Linux system, both 802.11 and 802.16 wireless network drivers will implement the SoftMAC interface, which will register them into SoftMAC control layer. The SoftMAC control layer is the only layer expose to the Linux Network Interface Layer. It will report the wireless network's status and transfer data packets to the upper layer of system. Figure 1.1 shows the Networking Architecture of SoftMAC.

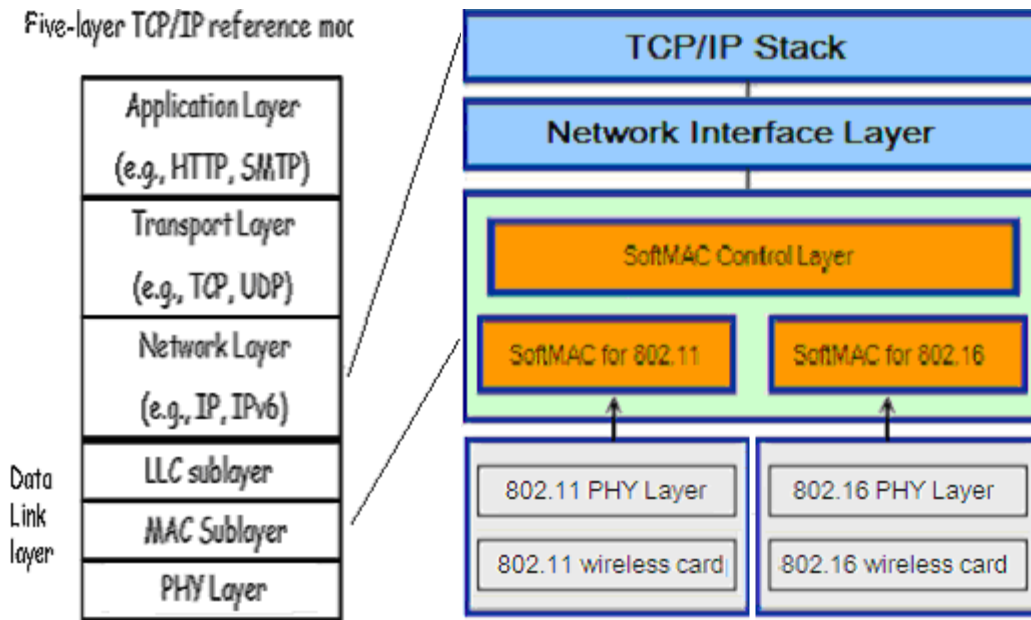


Figure 1.1 : Networking Architecture of SoftMAC

This SoftMAC architecture will allow researchers or users to query the status of registered wireless network and to control the data transmission through which wireless network. The extended SoftMAC is built on the top of existing MAC layer for each wireless network, and eliminates the significant different of wireless network MAC protocols.

2. Background

2.1 TCP/IP Model

The Open Systems Interconnection Reference Model (OSI Model) is a layered, abstract description for communications and computer network protocol design. From top to bottom, the OSI Model consists of the Application, Presentation, Session, Transport, Network, Data Link, and Physical layers [1]. A layer is a collection of related functions that provides services to the layer above it and receives service from the layer below it. Figure 2.1 displays all layers and functions in each layer.

OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation and encryption
		5. Session	Interhost communication
	Segment/Datagram	4. Transport	End-to-end connections and reliability (TCP)
Media layers	Packet	3. Network	Path determination and logical addressing (IP)
	Frame	2. Data link	Physical addressing (MAC & LLC)
	Bit	1. Physical	Media, signal and binary transmission

Figure 2.1 OSI Model [1]

TCP/IP is the environment that handles all these operations and coordinates them with remote hosts. The name TCP/IP refers to a suite of data communication protocols. Its name comes from the Transmission Control Protocol (TCP) and the Internet Protocol (IP). TCP/IP was created using the DoD (Department of Defense) model, which is made

up of four layers instead of the seven that make up the OSI model. TCP/IP standards are defined and described in Request for Comment (RFC) documents. The layers are as following:

Application layer

- Refers to standard network services like http, ftp, telnet as well as communication methods used by various application programs
- Also defines compatible representation of all data

Transport layer

- Manages the transfer of data by using connection oriented (TCP) and connectionless (UDP) transport protocols
- Manages the connections between networked applications

Internet layer

- Manages addressing of packets and delivery of packets between networks
- Fragments packets so that they can be dealt with by lower level layer (Network Interface layer Network)

Network Interface layer

- Delivers data via physical link (Ethernet is the most common link level protocol)
- Provides error detection and packet framing

TCP is Connection-Oriented protocol for communication between applications. When an application wants to communicate with another application via TCP, it sends a communication request. This request must be sent to an exact address. After a "handshake" between the two applications, TCP will setup a "full-duplex" communication between the two applications. The "full-duplex" communication will occupy the communication line between the two computers until it is closed by one of the two applications. UDP is very similar to TCP, but is simpler and less reliable.

IP is Connection-Less protocol for communication between computers. IP is "connection-less" as it does not occupy the communication line between two communicating computers. This way IP reduces the need for network lines. Each line can be used for communication between many different computers at the same time. With IP, messages (or other data) are broken up into small independent "packets" and sent

between computers via the Internet. IP is responsible for "routing" each packet to its destination.

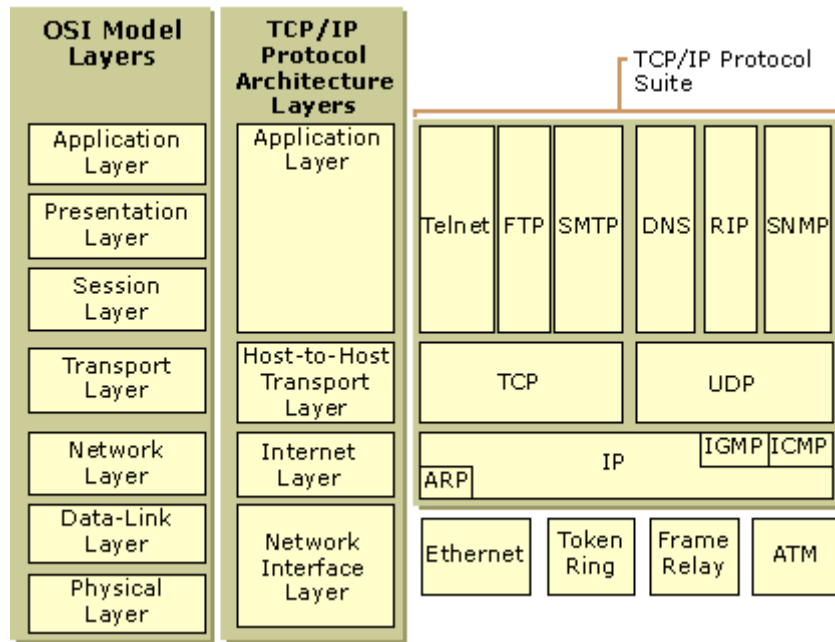


Figure 2.2 TCP/IP Model [1]

[2.2 IEEE 802.11 MAC Layer](#)

The IEEE 802.11 standard describes two types of wireless LANs, an ad hoc network and an infrastructure network. An ad hoc wireless LAN consists only of mobile stations. This type of wireless LAN is often set up for a very specific purpose, such as exchanging data during a single connection, and its lifetime is usually limited. An infrastructure network integrates mobile stations into a large network infrastructure through the use of access points (AP). The IEEE 802.11 MAC and MAC-management functions allow the mobile stations to find other mobile stations and APs, register with the wireless LAN, request encryption and power management services from the wireless LAN, and exchange data with other mobile stations and APs.

The basic access mechanism of 802.11 MAC is carrier sense multiple accesses with collision avoidance (CSMA/CA). The DCF (distributed coordination function) may be used in either the ad hoc or infrastructure wireless LANs. The DCF is quite similar to the CSMA with collision detection (CSMA/CD) used in IEEE 802.3 Ethernet. CSMA/CA

works by sensing the medium for activity before every transmission and deferring the transmission if the medium is active. As in 802.3, 802.11 uses a binary exponential backoff mechanism to spread transmission opportunities in time and minimize the likelihood of subsequent collisions.

The following summarizes primary 802.11 MAC functions, especially as they relate to infrastructure wireless LANs:

Scanning: The 802.11 standard defines both passive and active scanning. Passive scanning is mandatory where each NIC scans individual channels to find the best access point signal. Access Point periodically broadcast Beacons, which contain information about the Access Point, including the Service Set Identifier (SSID), supported data rates, etc. The NIC receives and use these Beacons to decide to use which AP.

Optional active scanning is that the mobile NIC initiates the process by broadcasting a Probe frame, and all Access Points will respond with a Probe response. Active scanning enables the NIC to receive immediate response from access points, without waiting for a beacon transmission.

Authentication: The 802.11 standard specifies two forms: Open system authentication and shared key authentication. Shared key authentication uses WEP (wired equivalent privacy) key for data communication. The radio NIC starts by sending an authentication request frame to the access point. The access point then places a text into the frame body of a response frame and sends it to the radio NIC. The radio NIC uses its WEP key to encrypt the text and then sends it back to the access point in another authentication frame. The access point decrypts the text and compares it to the initial text. If the text is equivalent, then the access point assumes that the radio NIC has the correct key. The access point finishes the sequence by sending an authentication frame to the radio NIC with the approval or disapproval.

Association: Once authenticated, the radio NIC must associate with the access point before sending data frames. Association is necessary to synchronize the radio NIC and access point with important information, such as supported data rates. The radio NIC initiates the association by sending an association request frame containing elements such as a SSID and supported data rates. The Access Point responds by sending an association response frame containing an association ID along with other information regarding the

access point. Once the radio NIC and Access Point complete the association process, they can send data frames to each other.

RTS/CTS: The optional request-to send and clear-to-send (RTS/CTS) function allows the access point to control use of the medium for stations activating RTS/CTS. The use of RTS/CTS alleviates the hidden node problem, that is, where two or more radio NICs can't hear each other and they are associated with the same access point.

Power Save Mode: The optional power save mode allows the radio NIC to be turn on or off to save battery power when there is no need to send data. In order to still receive data frames, the sleeping NIC must wake up periodically to receive regular beacon transmissions from the access point.

Fragmentation: The optional fragmentation function enables an 802.11 station to divide data packets into smaller frames to avoid needing to retransmit large frames in the presence of RF interference. The bits errors resulting from RF interference are likely to affect only a single frame. It requires less overhead to retransmit a smaller frame rather than a larger one.

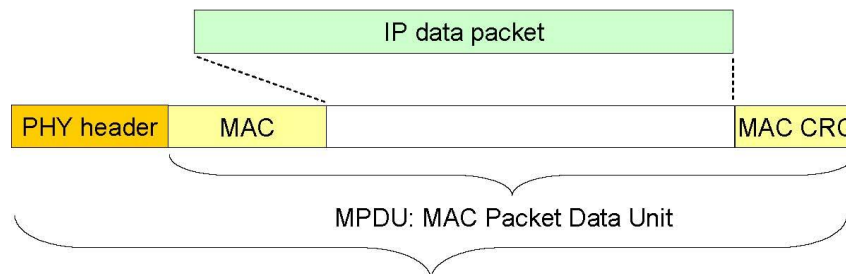


Figure 2.3 IEEE 802.11 Packet Format [1]

Figure 2.3 shows a 802.11 Packet in MAC layer. During 802.11 packet assembly, payload data from the IP layer, or the data that is being communicated, is encapsulated with MAC (media access controller) data and another four-byte segment of data that functions as a check sum (CRC or FCS). All of this data is assembled into an MPDU (MAC Packet Data Unit). When the packet is transmitted, the PHY layer appends a synchronization header.

802.11 PHY Layer

The infrared (IR) PHY uses baseband pulse position to transmit data at 1 and, optionally, 2 Mb/s. The frequency hopping (FH) PHY also provides data rates of 1 and, optionally, 2 Mb/s. This PHY was suitable for operation under the U.S. FH spread spectrum rules for the 2.4 GHz band designated for industry, scientific, and medical (ISM) applications; it remains usable under the rules as liberalized in 2000 to allow wideband FH spread spectrum systems. The 802.11 FH PHY provides 79 channels with a channel bandwidth of 1 MHz. For 1 Mb/s, the modulation used is two-level Gaussian frequency shift keying (GFSK) with a nominal bandwidth bit period of 0.5.

The direct sequence (DS) PHY, as extended in the 802.11b amendment, provides data rates of 1, 2, 5.5, and 11 Mb/s. The modulation used for these rates is differential binary phase shift keying (DBPSK) and differential quadrature phase shift keying (DQPSK), respectively. The 5.5 and 11 Mb/s data rates use complementary code keying (CCK) as the spreading mechanism.

The orthogonal frequency division multiplexing (OFDM) PHY, described in 802.11a, provides eight data rates: 6, 9, 12, 18, 24, 36, 48, and 54 Mb/s. It uses binary phase shift keying (BPSK), quadrature phase shift keying (QPSK), 16-QAM (quadrature amplitude modulation), and 64-QAM modulation schemes coupled with forward error correction coding of rates 1/2, 2/3, and 3/4.

2.3 IEEE 802.16 MAC Layer

The IEEE 802.16 MAC protocol was designed for point-to-multipoint broadband wireless access applications. It addresses the need for very high bit rates, both uplink (to the BS) and downlink (from the BS). Access and bandwidth allocation algorithms must accommodate hundreds of terminals per channel with the possibility of multiple end users sharing each terminal. The services required by these end users are varied in their nature and include legacy time-division multiplex (TDM) voice and data, Internet Protocol (IP) connectivity, and packetized voice over IP (VoIP). To support this variety of services, the 802.16 MAC must accommodate both continuous and bursty traffic. Additionally, these services expect to be assigned QoS in keeping with the traffic types.

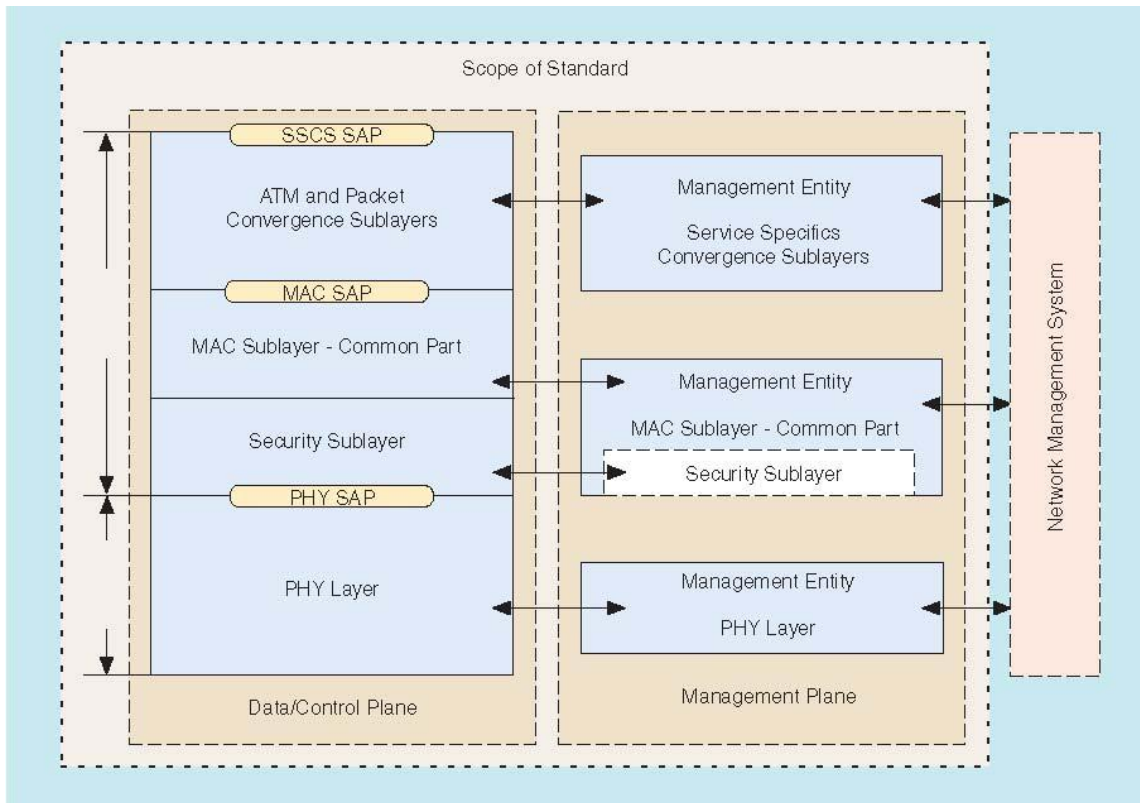


Figure 2.4 IEEE 802.16 Reference Model and Protocol Stack [2]

The 802.16 MAC protocol was designed for point-multi-point broadband wireless access. Access and bandwidth allocation algorithms accommodate hundreds of user terminals per single channel. User terminals may also be shared among many end-user equipments like phones and PCs. To support variety of services, 802.16 needs to accommodate bursty and continuous traffic, with required QoS of every service. On downlink data is multiplexed with TDM. Uplink is shared with TDMA. 802.16 is connection-oriented. All services, even those inherently connectionless, are mapped to a connection. It provides mechanism for required bandwidth, associating Grade of Service (GoS) and traffic parameters and for transporting and routing data to the appropriate sublayer. Connections are referred with 16-bit Connection Identifiers (CID) and may require continuously bandwidth or band-on-demand. Upon entering the network, three management connections are established, in both directions. Every connection is used for different QoS connection type:

- basic connection – used to transfer of short, time critical MAC and RLC messages

- secondary management Connection – used for transfer of standard-based protocols such as DHCP,TFTP,SNMP
- Other types of connection, like connection reserved for broadcasting

The 802.16 MAC protocol must also support a variety of backhaul requirements, including both asynchronous transfer mode (ATM) and packet-based protocols.

Convergence Sublayers are used to map the transport-layer-specific traffic to a MAC that is flexible enough to efficiently carry any traffic type. Through such features as payload header suppression, packing, and fragmentation, the convergence sublayers and MAC work together to carry traffic in a form that is often more efficient than the original transport mechanism.

802.16 PHY Layer

802.16 Physical layer was defined for a wide range of frequency from 2 up to 66 GHz. The sub-range 10-66 GHz, there is an assumption of Line-Of-Sight propagation. In this scheme single carrier modulation was chosen, because of low complexity of the system. The downlink channel is shared among users with TDM signals. Subscriber unit are being allocated individual time slots. Access in uplink is being realized with TDMA. Channel bandwidths are 20 or 25 MHz in USA and 28MHz (Europe). Duplex can be realized with either TDD or FDD scheme. In the 2-11 GHz bands communication can be achieved for licensed and non-licensed bands. The communication is also available in NLOS conditions.

The 802.16a Draft3 air interface specification describes three formats:

- Single Carrier modulation (SC)
- OFDM with 256 point transform
- OFDMA with 2048 point transform

The Forward Error Correction (FEC) is used with Reed-Salomon Codes GF(256). It is also paried with inner block convolutional code to robustly transmit critical data, like Frame Control or Initial Access.

One aspect of WiMAX QoS provisioning is a grant-request mechanism for letting users into the network. This mechanism's operation and value become apparent from a comparison of WiMAX with the CSMA/CD or CSMA/CA mechanisms used in LAN

technologies such as 802.11. When a CSMA/CA based wireless LAN has fewer than 10 users per access point, the network experiences little contention for use of airtime. Occasional packet collisions occur, and they require back-off and retransmissions, but the resulting overhead does not waste a significant amount of bandwidth. If the number of CSMA/CA accesspoint users goes up to dozens or hundreds of users, many more users tend to collide, back-off and retransmit data. In such an environment, average network loading factors can easily rise past 20 to 30 percent. Thus, users will notice delays—especially in streaming-media services. WiMAX avoids such issues by using a grant-request mechanism that allocates a small portion of each transmitted frame as a contention slot. With this contention slot, a subscriber station can enter the network by asking the base station to allocate an uplink (UL) slot. The base station evaluates the subscriber station's request in the context of the subscriber's service-level agreement and allocates a slot in which the subscriber station can transmit (send UL packets).

2.4 Linux Network Device Driver

Linux operating system is based on a monolithic kernel. The network drivers are compiled into the kernel space. The Linux kernel is inherently considered efficient and secure for networking. In Linux, the logical-link control (LLC) layer is implemented in the operating system kernel with network adapters connected to the operating system kernel by the network devices. Figure 2.5 illustrates the architecture of the Linux Network.

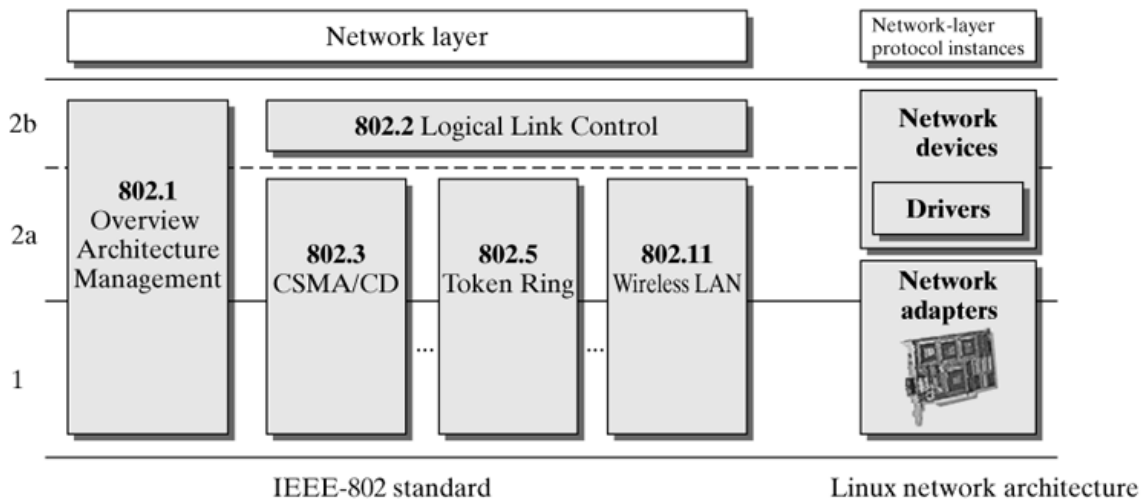


Figure 2.5 Linux Network Architecture [3]

Modularity and granularity of recent Linux kernel, provides a concise well-organized, efficient and a solely higher layer protocol independent coding interface. This enables a programmer to develop a network device driver as kernel module, instead of part of the monolithic kernel. As a kernel module, the WLAN chipset driver, request resources needed for the operations of the device such as I/O port, interrupt (IRQ) number etc. The kernel maintains a global list of detected network devices. Each interface defined by struct `net_device`, declared in `/include/linux/netdevice.h`. The `net_device` structure forms the basis of each network device in the Linux kernel. It contains not only information about the network adapter hardware (interrupt, ports, driver functions, etc.), but also the configuration data of the network device with regard to the higher network protocols (IP address, subnet mask, etc.). The `net_device` structure has the following information:

- General Fields of a Network Device
- Hardware-Specific Fields
- Data on the Physical Layer
- Data on the Network Layer
- Device-Driver Methods

This represents the general interface between higher protocol instances and the hardware used. It also allows you to abstract from the network components used. For an efficient implementation of this abstraction, we once again use the concept of function pointers. For this reason, the `net_device` structure contains a number of function pointers,

which are called by higher protocols by using their global names, and then the hardware-specific methods of the driver are called from each network device.

Whenever a device driver register itself, using `register_netdevice()`, it initializes the hardware and allocate the resources it needs by filling up `net_device`'s items. The following methods are common for each network device interface in `init_module()` when a driver as kernel module is loaded into the kernel.

2.5 MADWiFi and SoftMAC

Multiband Atheros Driver for WiFi (MADWiFi) is an open-source Linux driver for 802.11 a/b/g universal NIC cards - Cardbus, PCI, or miniPCI - using Atheros chip sets. MADWiFi is one of the most advanced WLAN drivers available for Linux today. It is stable and has an established userbase. The driver itself is open source but depends on the proprietary Hardware Abstraction Layer (HAL) that is available in binary form only.

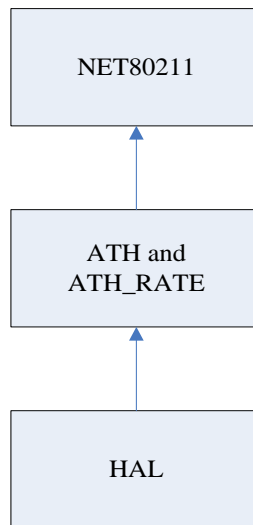


Figure 2.6 Architecture of Atheros NIC driver (MADWiFi)

Atheros uses a "hardware abstraction layer" or HAL to provide a common hardware interface for operating systems. The HAL is written in the machine code of the computer hosting the wireless card, and abstracts common functionality across different individual chipsets. Although the HAL is distributed in binary-only format and not extensively documented, there have been attempts to produce an "open-source" HAL. We have only used these open-source references during development.

The current MADWiFi driver supports multiple APs and concurrent AP/Station mode operation on the same device. The devices are restricted to using the same underlying hardware, thus are limited to coexisting on the same channel and using the same physical layer features. Each instance of an AP or station is called a Virtual AP (or VAP). Each VAP can be in either AP mode, station mode, “special” station mode or monitor mode. Every VAP has an associated underlying base device, which is created when the driver is loaded. The MADWiFi driver supports a wireless extension kernel API that allows configuring the device using common wireless tools like ifconfig and iwconfig. A rich supported operational modes such as station, i.e. managed mode, Access Point, i.e. master mode, ad-hoc mode i.e. IBSS mode, WDS (wireless distributed system) to create large wireless network by linking with neighbor AP, monitor mode etc. made MADWiFi platform complex, not easily understandable and has been cumulated lots of code.

The whole MADWiFi code is consists of several parts. The device driver, as kernel module named ath_pci consists from modules ath, ath_HAL, net80211 and ath_rate.

- net80211 or ieee80211 stack

net80211 or ieee80211 stack is originally hacked from FreeBSD which contains generic IEEE802.11 functionality. For BSDs, this stack supports numerous WLAN devices. It, however, has been imported and customized only for Atheros wireless LAN chipsets. This module implements lost of called back which can be called by ath_HAL, ath module provided that, it has to be exported by EXPORT_SYMBOL. net80211 module also consists of WLAN authentication, cryptographically part.

- ath module

ath module defines Atheros WLAN controller specific callbacks for net80211 module access to the hardware through HAL module. It contains time critical part of 802.11 management, e.g. beacon management, device’s ioctl, configure and setup transmit (TX) and receive (RX) queue, PCI bus controlling connected with the CPU etc.

- ath_HAL module

HAL module, Hardware Access Layer, is responsible to access to hardware. This closed source component, basically maintained by the vendor, Atheros, itself, can be thought something like firmware of card with the only exception is, its not stored into the card, instead consider as kernel module. Commercial point is, it’s required less flash

memory on the board, which can reduce market value of device. By definition, HAL is not exactly firmware, since firmware is hardware program executable on board microcontroller. According to the argument of vendor, due to chipset's versatility to tune wide range of frequencies, even in unlicensed bands (non-ISM), to enforce limit on transmit power etc and for some legal issue, Atheros keeps the code of HAL module as closed source. Moreover, there is no documentation for HAL except a public interfaces in HAL/ah.h. Soon we will see this unavailability has made our implementation so hard.

- ath_rate module

ath_rate module selects the appropriate algorithm for the best transmission rate. Among 802.11a, b, g, multiple bit rates, this module sets the device's transmission when sending data packet. MADWiFi includes three different algorithms to choose bit rate:

- a) onoe algorithm
- b) amrr algorithm
- c) SampleRate algorithm.

SoftMAC is proposed and developed by University of Colorado based on MADWiFi and is for Atheros chipset only. It is an open source project to allow researchers to easily construct and deploy dynamic MAC layers on system running Linux. The algorithms and codes can be found and download from Internet, but as all open-source projects, the detailed document and support of SoftMAC is not described.

The key point of the SoftMAC system is to precisely control the content and timing of wireless transmission and reception. As an extension for SoftMAC system, Multi-MAC is introduced. It allows researchers to implement multiple MAC protocols simultaneously to achieve best network performance. Multi-MAC chooses the MAC layer capable of achieving the best performance while ensuring that incoming frames are decoded using the correct MAC layer algorithm.

3. Theory Approaches

The popularly deployed 802.11-based WLAN and its wide availability hardware facility make it first choice for many users. 802.16 wireless network has a better LOS (line of sight) capacity and QoS support. When both wireless networks are available, suppose that we have to do the switch or handoff between the two wireless networks, there are different ways to solve the problem.

1. Transportation or Routing layer application

It is easy to implement. But it still can't solve 802.11 MAC problems.

2. New MAC protocol

802.11 and 802.16 have a total different MAC layer. To design a new MAC protocol, it is too complicated, too expensive and too hard to implement.

3. SoftMAC: Layer 2.5 MAC Proposed by Microsoft

This solution has a low cost and easy to implement on current MAC layer, however, it is tied to Microsoft Windows and only limited to 802.11 MAC interface. More importantly, we don't have the source code to research and thus no ability to do any possible modification.

4. SoftMAC: Proposed and developed by University of Colorado

This solution has a lower cost and easy to implement on current MAC layer, easy to modify and integration with higher layers. It allows researchers to easily construct and deploy dynamic MAC layers on system running Linux. More important, it is an open-source project.

SoftMAC is a software controlled platform between MAC layer (or Ethernet MAC layer) and IP layer to realize some control function on content and scheduling of transmission to improve network performance.

SoftMAC, first is proposes by MS Research Center, designed Layer 2.5 SoftMAC objective is to coordinate the Best-effort and multi-streaming traffic to improve QoS of multimedia service as audio/ video service over multi-hop wireless Network and Ethernet and WLAN combined typical home network. [14]

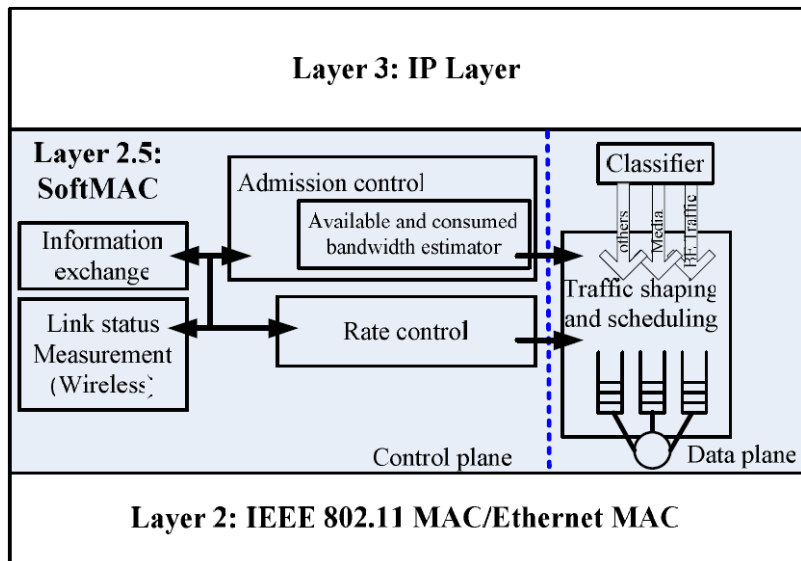


Figure 3.1 Layer 2.5 SoftMAC architecture

As shown in Figure 3.1, this SoftMAC, working between the MAC layer and the network layer has two planes: a control plane and a data plane. Depending on the decision the control plane has made, the data plane implements corresponding functions. The core mechanism of this implementation is

1. Admission Control for regulating the load RT traffic
2. Rate Control to minimize the effect BE traffic on real-time traffic
3. Non-Preemptive Queuing to provide high priority to real time traffic.

Second, the SoftMAC and extended Multi-MAC proposed and developed by University of Colorado, it has developed a software system for researchers to easily construct and deploy experiment on MAC protocol. SoftMAC system uses a commodity 802.11a/b/g network card with a chipset manufactured by Atheros Corporation to build a software radio with predefined physical layers and flexible MAC layer. [9]

The key point of the SoftMAC system is to precisely control the content and timing of wireless transmission and reception. Specifically, this is realized by performing following 6 tasks:

1. Override 802.11 MPDU frame format
2. Eliminate automatic ACK and retransmission
3. Eliminate RTS/CTS exchange

4. Eliminate virtual carrier sense (NAV)
5. Control PHY Clear Channel Assessment (CCA)
6. Control transmission backoff.

As an extension for SoftMAC system, MultiMAC is introduced [10]. It allows researchers to implement multiple MAC protocols simultaneously to achieve best network performance. Multi-MAC chooses the MAC layer capable of achieving the best performance while ensuring that incoming frames are decoded using the correct MAC layer algorithm. The decision is made to specific node and traffic flow. For example, when the traffic is busy, TDMA is used in MAC layer, while in low-contention periods, it uses CSMA/CA MAC protocol with Reed-Solomon coding to reduce the bit-error rate at the expense of extra coding and CPU overhead. The interface of Multi-MAC connecting Linux Kernel Network stack and physical emphasize three aspects of MAC protocols- reception decoding, transmission encoding and transmission timing.

Before above SoftMAC is proposed, Rao and Stoica in Berkeley proposed the concept of Overlay MAC Layer [8], which can not only provide better flexibility but improving fairness, throughput and predictability as well. The overlay design and implementation focus more on best-effort traffic. And [11] introduces a systematic and automatic method to dynamically combine several MAC protocols into a single higher layer. The new approach makes it possible to achieve best performance by choosing most appropriate MAC protocol without knowing the changing and unpredictable network conditions. It runs without any centralized control or exchange of messages but only with local network feedback information.

From the discussion above, we can draw the conclusion. To connect 802.11 and 802.16 in MAC layer, SoftMAC from Colorado is the better choice for us to do the project.

To extend the ability of the SoftMAC, a standard SoftMAC interface is defined. We add an additional layer on the top of existing MAC layer, which is regardless the significant difference of different wireless network protocols. So we have created a software architecture that permits researchers to easily construct and deploy experimental

dynamic MAC layers on systems running Linux. By implement the SoftMAC on top of 802.11 and 802.16 MAC layer, we can connect the system to the two wireless network and achieve the switch over functions. See Figure 1.1 for Networking Architecture of SoftMAC

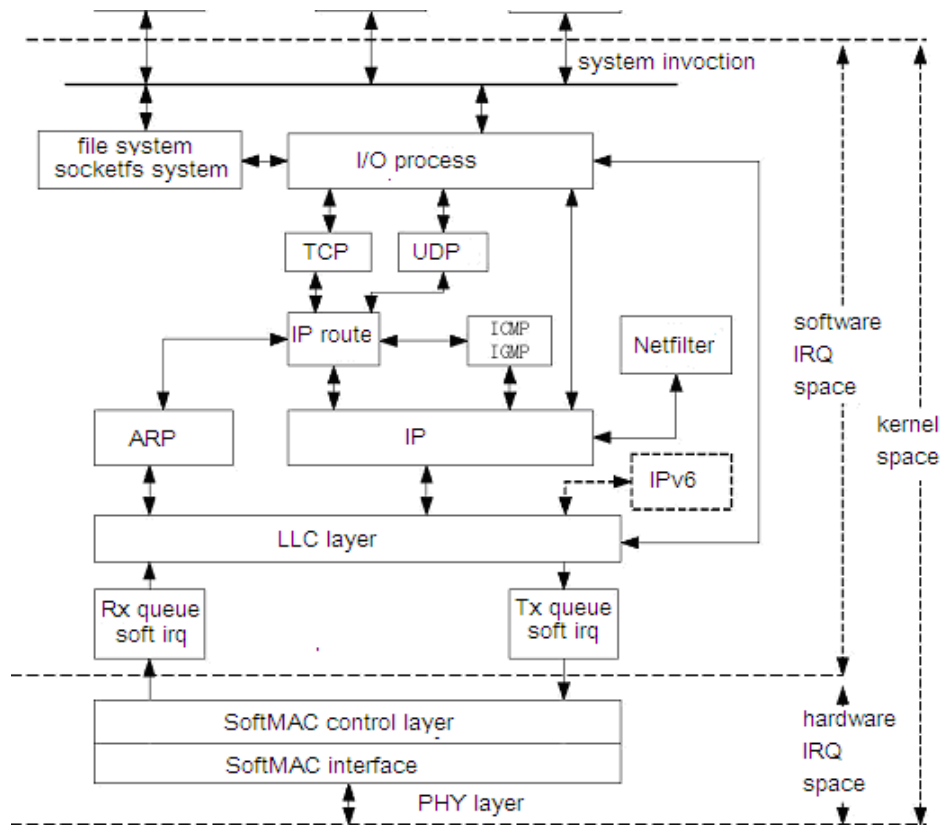


Figure 3.2 SoftMAC in Linux Network Architecture

SoftMAC is an abstraction designed to facilitate the creation of custom and experimental MAC layers when used with Software Defined Radio hardware. A traditional network interface system is structured with a tightly coupled MAC and PHY layer exporting a standard API to the operating system. The SoftMAC breaks out the integrated MAC and PHY layer into separate (but still coupled) components. Our SoftMAC is designed running on Linux platform. To better understand the Linux network system, Figure 3.2 shows SoftMAC in Linux Network Architecture.

The standard SoftMAC interface will enhance the native Linux *net_device* structure. Both 802.11 and 802.16 drivers will inherit from this interface. So SoftMAC control layer can have a standard way to access the status of the each sub network, and control the packet transmission of each wireless network.

This architecture affords greater flexibility than the traditional network stack when used with appropriately capable hardware. Most MAC layers are still implemented either wholly or partially by the underlying hardware and are immutable. With such hardware the SoftMAC abstraction makes no sense, and serves only as an extra layer of inefficiency. However, recent advances in Software Defined Radio technology have resulted in much more flexible network hardware. Fully exploiting the capabilities of this hardware requires a more flexible network stack. SoftMAC is designed to strike a balance between flexibility and efficiency that will permit experimentation with this hardware.

4. Platform and Implementation

The SoftMAC in Heterogeneous Wireless Network project is funded by The Institute for Information Industry (III). In this project, a 802.11 WiFi wireless network card and a 802.16 WiMAX wireless network card are used. The system is required to run on an IXP425 test board. From Figure 4.1, we can see the relationship between the 802.11 SoftMAC driver and 802.16 SoftMAC driver and the SoftMAC control layer.

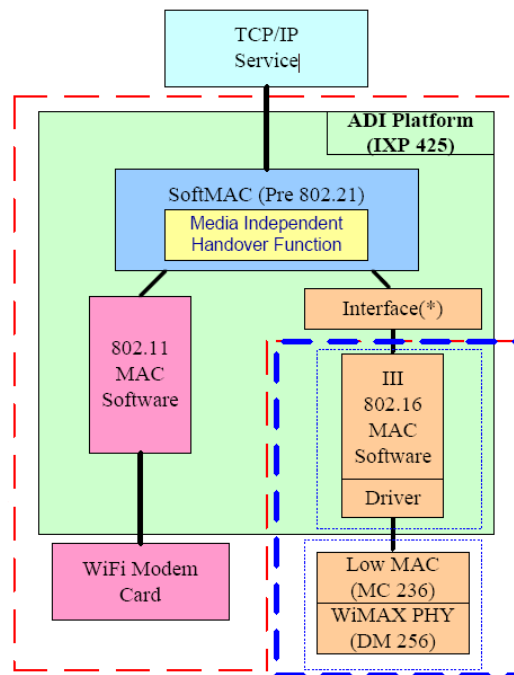


Figure 4.1 SoftMAC with 802.11 and 802.16

The SoftMAC interface is defined and compiled into the 802.11 or 802.16 wireless network card's driver. For 802.11, MADWiFi, the open source driver is used, and the 802.11 NIC is Atheros AR5212. For 802.16, there is no open source driver that can be used. Our partner provides a NIC, which is limited to use a driver on Linux Kernel 2.4.22. The SoftMAC control layer will communicate with both 802.11 and 802.16 SoftMAC interface. It can query the status of each wireless connection, or get the SNR of each individual wireless network. By the algorithm, the SoftMAC control layer will select the better link to transfer or receive packets.

4.1 IXP425 Test Board

The system is implemented on an IXP425 test board. The IXP425 network processor is a highly integrated, versatile single-chip processor that can be used in a variety of products requiring network connectivity and high performance to run their unique software applications. The Intel IXP425 network processor combines integration with support for multiple WAN and LAN technologies in a common architecture to meet requirements for high-end gateways, Voice over IP (VoIP) applications, wireless access points, SME routers, switches, security devices, (DSLAMs), xDSL line cards, industrial control and networked imaging applications. The IXP425 is an implementation of the ARM compliant, Intel XScale microarchitecture combined with communication peripherals including, 2 high speed Ethernet MACs, hardware accelerated cryptography, 2 high speed serial ports, a local PCI interface and DMA controller.



Figure 4.2 IXP425 test board

The IXP425 network processor's primary features (as used in Pronghorn SBC) are:

- Intel XScale Core running at 266 MHz or 533 MHz (depending on SBC model)
- Three NPEs for Layer-2 packet/frame network processing.
- Two 10/100-Mbps, full-duplex IEEE-802.3 MAC's with MII interface (not all SBC models support two actual Ethernet ports)

- Dedicated SDRAM with 32-bit memory interface operating at 133 MHz (equal to system clock frequency)
- Expansion Bus
- Two UARTS
- PCI 2.2 bus:
- GPIOs

IXP425 network processor has a RISC instruction set. Like other RISC architectures, ARM processors are designed to efficiently access 'aligned data' - i.e. words which lie on addresses that are multiples of 4 and halfwords which lie on addresses that are multiples of 2. Such data is located on its natural size boundary. ARM's compilers normally align global variables to these natural size boundaries so that these items can be accessed efficiently using the LDR/STR instructions. This contrasts with most CISC architectures where instructions are available to directly access 'unaligned data'. This means that care needs to be taken when porting legacy code, which carries out, such unaligned accesses from such architectures to the ARM. The following differences have to be concerned during the development and porting:

- Unaligned Pointers
- Unaligned fields in structures
- Unaligned LDR for accessing halfwords

The ARM compilers expect normal 'C' pointers to point to an aligned word in memory, as this allows the compiler to generate more efficient code. Thus if you wish to define a pointer to a word that can be at any address (i.e. that can be at a non-natural alignment) then you must specify this using the `__packed` qualifier when defining the pointer:

```
__packed int *pi; // pointer to unaligned int
```

The ARM compilers will not then use an LDR, but instead generate code which correctly accesses the value regardless of the alignment of the pointer. This code generated will be a sequence of byte accesses, or variable alignment-dependent shifting and masking (depending on the compile options) and will therefore incur a performance and code size

penalty. Porting code and detecting unaligned accesses is a big headache in the whole project.

4.2 SoftMAC for MADWiFi Driver

MADWiFi driver lives in Linux kernel space, is for WLAN device based on Atheros WLAN chipset. In the project, we have selected WLAN card based on Atheros AR5212 chipset which supports 802.11a/b/g. Atheros WLAN chipset has multi-protocol MAC or baseband processor supports Radio-on-Chip (RoC) that can operate dual band 2.4/5GHz. The radio modem use OFDM in 5 GHz band in 8 different channels with throughput of up to 54Mb/s rates (depending of range).

MADWiFi is an Open Source Linux kernel device driver for Wireless LAN chipsets from Atheros. Since MADWiFi is an open source code, modifications to the existing coding and implementations of new features can be performed without restrictions. This driver will be installed on all four computers in the test bed. The driver works such that the WLAN card (D-link WDA 1320) will appear as a normal network interface in the system. Additionally, there is support for the Wireless Extensions API. The API provides different operational modes such as client station, access point, network monitor, and ad-hoc mobile station. For this design project, one of the computers is configured as the client station, one as a dummy internet server while the other two computers are configured as access points.

In order to support the WiMAX driver provided by our partner, all of the computers must have Linux Kernel 2.4.x (Redhat 9) series installed in them. The items in Table 1 must also be supported in the Linux kernel.

Crypto support
Wireless Extensions support
CONFIG_NET_DIVERT

Table 1: MADWiFi Required for Linux Kernel compile.

After installing MADWiFi, the steps to configure the driver are documented in Table 2.

Steps	Command Code
1. Loading MADWiFi Module	<i>modprobe ath_pci</i>
2. Creating an Interface for client station	<i>wlanconfig atho create wlandev wifi0 wlanmode sta</i>
3. Creating an Interface for Access Point	<i>wlanconfig atho create wlandev wifi0 wlanmode ap</i>
4. Initialize all wireless interface in each computer	<i>Ifconfig ath0 up</i>
5. Insert the AP scanning module into the client mobile station	<i>modprobe wlan_scan_sta</i>
6. Scan for the AP from client	<i>wlanconfig ath0 list scan</i>
7. Connect to AP1	<i>iwconfig ath0 essid "AP1"</i>
8. Obtain IP address without DHCP	<i>Ifconfig ath0 <IP address> netmask <netmask> up</i>

Table 2: Procedure for MADWiFi Driver Configuration.

We have to modify the MADWiFi driver to support SoftMAC Layer. SoftMAC PHY layer is embedded in the MADWiFi driver to support the SoftMAC control layer functions. Table 3 is part of the modification of MADWiFi driver.

SoftMAC for MADWiFi	
struct ath_softc {	
struct net_device	sc_dev; /* NB: must be first */
struct net_device	sc_rawdev; /* live monitor device */
struct semaphore	sc_lock; /* dev-level lock */
struct net_device_stats	sc_devstats; /* device statistics */
struct ath_statssc_stats;	/* private statistics */

```

...
...

/* SoftMAC PHY*/
int          sc_cu_softmac;
int          sc_cu_softmac_alwaystintr;
int          sc_cu_softmac_phocus_settletime;
int          sc_cu_softmac_phocus_enable;
u_int32_t    sc_cu_softmac_phocus_state;
u_int8_t     sc_cu_softmac_wifictl0;
u_int8_t     sc_cu_softmac_wifictl1;
int          sc_cu_softmac_noautocalibrate;
int64_t      sc_cu_softmac_zerotime;
atomic_t     sc_cu_softmac_tx_packets_inflight;
struct tq_struct  sc_cu_softmac_worktq;
u_int32_t    sc_cu_softmac_txlatency;
u_int32_t    sc_cu_softmac_options;
CU_SOFTMAC_MACLAYER_INFO *sc_cu_softmac_mac;
CU_SOFTMAC_MACLAYER_INFO
*sc_cu_softmac_defaultmac;
CU_SOFTMAC_PHYLAYER_INFO *sc_cu_softmac_phy;
int sc_cu_softmac_phy_id; /* instance id */
struct cu_softmac_athmac_instance *sc_cu_softmac_mac_inst;
struct list_head sc_cu_softmac_procf_data;
struct list_head sc_cu_softmac_phy_list;
struct list_head sc_cu_softmac_mac_list;
};

```

Table 3: SoftMAC for MADWiFi.

4.3 SoftMAC Layer Structure Design

The source code of the SoftMAC is compiled and loaded into the /proc file system in the Linux kernel. Table 4 shows part of the structure for SoftMAC Control Layer. It is the key part of the project. All handover function will be implementing here. It will communicate with SoftMAC PHY layer in MADWiFi driver to get enough information to do the algorithm. Now only a simple handover algorithm is used.

SoftMAC Layer Info

```
typedef struct CU_SOFTMAC_LAYER_INFO_t {
    /* @brief Layer name. */
    char name[CU_SOFTMAC_NAME_SIZE];

    /* @brief Create a new instance of this layer type */
    void (*cu_softmac_layer_new_instance)(void*);

    /* @brief Destroy an existing instance of this layer type */
    void (*cu_softmac_layer_free_instance)(void*, void *inst);

    /* @brief Private data */
    void *layer_private;
    ....
    ....

    /* softmac internal */
    struct hlist_node name_hlist;

    /* softmac query funtions */
    float (*snr)(const void*); /* Signal-to-noise ratio*/
    float (*packet_loss_statistics)(const void*); /* packet
loss ratio*/
    int (*status_of_MACs)(const void*); /* status of MAC */
};
```

Table 4: Structure of SOFTMAC_LAYER_INFO.

For each Network driver in Linux, it has to be embedded the functions to support by SoftMAC control. In the SoftMAC Control layer, we can call the functions defined in SoftMAC PHY layer to get these values.

1. SNR/CINR,

The register in MC236 has provided the 'SNR/CINR' value. However, the type is unsigned integer which follows the definition in 802.16 specification. (unit : 0.25dBm)

2. Packet Error rate,

There is not such field to describe the value. But there are 3 values that we can use to calculate the PER :

(1) Number of Rx-Msg : the number of PDU put in Rx-Data-FIFO. (HW)

(2) Number of CRC-Error : the number of dropped PDU due to CRC-error

(3) Number of Symb-Error : the number of dropped PDU due to at least one symbol error.

Therefore,

$$PER = (2) + (3) / ((1) + (2) + (3))$$

3. Status of MAC (active / inactive)

4. Packet data Tx/Rx between softmac and 802.16 interface

Table 5 has the Script to load SoftMAC to Linux Kernel.

Command Code
<pre> ifconfig ath0 up ifconfig ath0 down iwconfig ath0 mode monitor iwconfig ath0 channel 5 iwconfig ath0 essid SoftMAC97 ifconfig ath0 up insmod SoftMAC_netif insmod SoftMAC_rslib insmod SoftMAC_multimac echo athphy > /proc/SoftMAC/create_instance echo 1 > /proc/SoftMAC/insts/athphy0/SoftMAC_enable </pre>

Table 5: Scripts to load SoftMAC.

Table 6 shows how to load MultiMAC by using the SoftMAC to send data to different destinations.

Steps	Command Code
1. Create a SoftMAC instance	<i>echo multimac ></i> <i>/proc/SoftMAC/create_instance</i>
2. Allocate the MAC layer resources to SoftMAC	<i>echo multi1 ></i> <i>/proc/SoftMAC/insts/athphy0/mac_layer</i>
2. Add each MAC layer resources to SoftMAC	<i>echo formagemac ></i> <i>/proc/SoftMAC/insts/multi1/addmaclayer</i>
3. Set the computer as client mode	<i>#iwconfig ath0 mode client</i>

Table 6: Procedure for MultiMAC Configuration.

4.4 Handover Algorithm

There are a number of parameters associated with the functionality of the handover process. These parameters are defined as constants and may be manipulated by the user to tailor the handover constraints necessary for the desired application. Each parameter is associated with a specific constraint, which are detection, power, and throughput. Detection type parameters store constants that relate to the detection of the intelligent handover scheme. Power type parameters are used as decision parameters to aid the determination of an effective handover by using SNR. Throughput parameters use bandwidth as the decision comparison. This work is done in “Seamless Wireless Integration”, by James Ho, Christie Kong. [14] Table 7 summarizes the parameters and their corresponding purposes.

Type	Parameter	Purpose
Detection	T_drop	Duration of time under threshold before connection dropout
	Refresh_time	Duration between optimal connection determination
	maximum_cxns	Number of simultaneously active connections
Power	power_threshold	Minimum SNR threshold to maintain connection
	power_enable	Flag to optimize connection based on SNR
Throughput	thru_threshold	Minimum bandwidth threshold to maintain connection
	thru_enable	Flag to optimize connection based on bandwidth

Table 7: Required Parameters for Intelligent Handover Algorithm [14]

To summarize the handover process implemented in the admission control block, the generic algorithm for the handover algorithm is outlined in Figure 6.

```
refresh_counter = 0;
while (true)
{
```

```

refresh_counter++;
if ( refresh_counter == refresh_time )
{
refreshcounter = 0;
list = detect_environment();
max_snr = -1;
max_bw = -1;

foreach (list)
{
if ( power_enable == true && power > power_threshold)
{
if ( current_snr > max_snr )
{
max_snr = current_snr;
best_cnx = this;
}
}

if ( thru_enable == true && thru > thru_threshold)
{
if ( current_bw > max_bw )
{
max_bw = current_bw;
best_cnx = this;
}
}
}
}
}

```

Figure 4.3: Algorithm for the Handover Algorithm. [14]

The above algorithm serves as the intelligent handover scheme for the prototype demonstration. The refresh timer is a predefined variable set for duration between optimal connection determinations. In our demonstration, 1 second is used as the value of refresh timer. Therefore, SoftMAC will detect the available access point in the near environment in every 2 seconds. In our algorithm, there are two factors to optimize the connection which are SNR and throughput. Every time when SoftMAC go over the access point

table, it will check if the current access point's signal strength exceeds the threshold value and the current best signal strength. If it does, it will replace the current access point with this access point. Therefore, our client station will always be able to connect or handover to the best access point available before losing any connection.

4.5 Experimental Results

The purpose of building a new network layer (SoftMAC) is to perform seamless handover between IEEE 802.11 and IEEE 802.16 access points. Due to the limitation of the 802.16 Linux driver and the absent of the 802.16 access point, the whole system can not be tested and demonstrated in a 802.11 and 802.16 heterogeneous wireless network. But the theory, the model and the SoftMAC coding can be tested in a system with two 802.11 wireless network cards installed.

This test and demonstrate is recorded in “Seamless Wireless Integration”, by James Ho, Christie Kong. [14] It summarizes the deviations from the original design functionality and discusses any design constraints that were not met. For the design specifications that were met, the novelty of the design and how much the design exceeded the requirement specifications are discussed.

Demonstration of the SoftMAC Handover

The handover procedure is demonstrated using one desktop terminal installed with MADWiFi and SoftMAC as well as three more wireless desktop stations with MADWiFi. The test bed consists of two stations working as access points/bridge, one station working as a dummy internet server and one station with SoftMAC working as a client node. All desktops are running on RedHat Linux 2.4.21. The framework is shown in Figure 4.4.

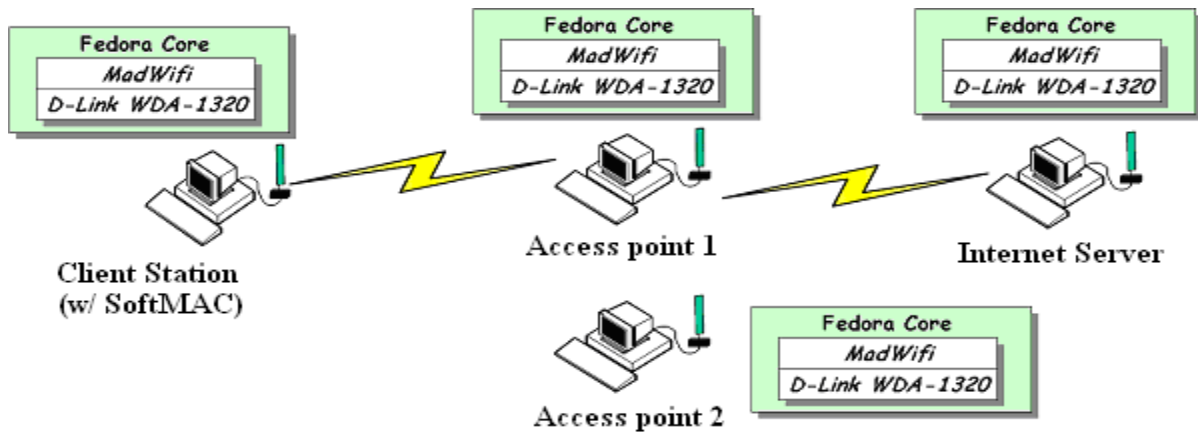


Figure 4.4: Seamless Handover Test-Bed. [14]

Each station is installed with two 802.11 wireless card in order to maintain a seamless connection. Figure 4.5 below shows the IP configuration of the two wireless cards in the client station. It shows that they have a different IP address and that they are used to connect to one of the access point separately. They both communicate with the multiMac layer in SoftMAC to decide which access point to connect to.

```
[root@localhost script]# ifconfig
ath0      Link encap:Ethernet  HWaddr 00:1B:11:C7:1A:1E
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:200
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5 Memory:f8a42000-f8a52000

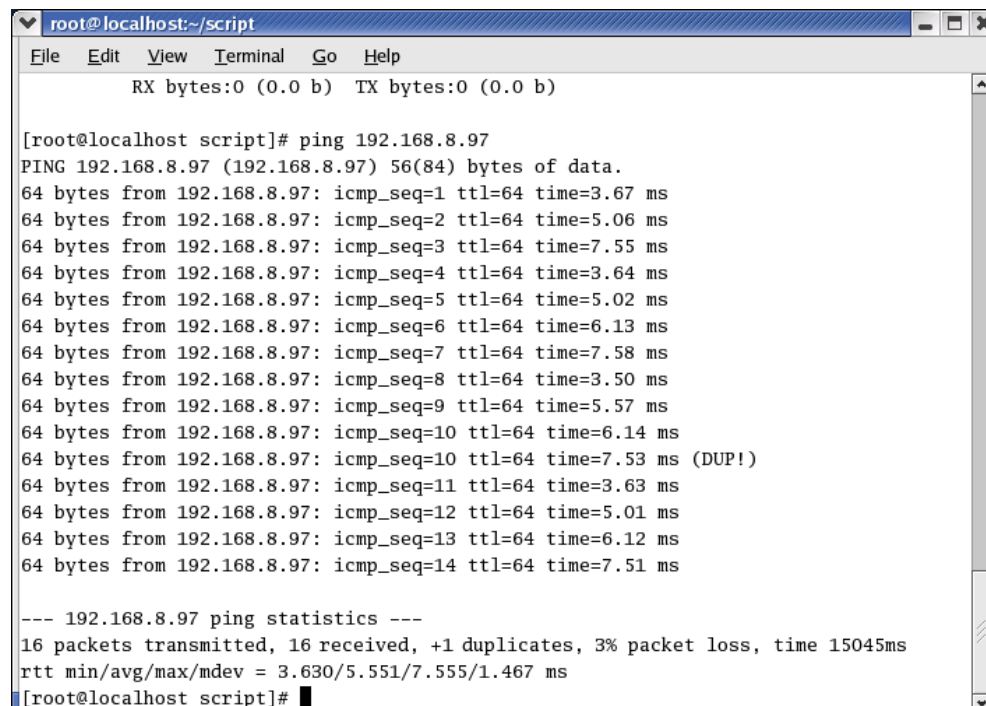
multi1    Link encap:Ethernet  HWaddr 00:7F:22:25:F5:19
          inet addr:192.168.8.99  Bcast:192.168.8.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

multi2    Link encap:Ethernet  HWaddr 00:FF:85:25:E1:79
          inet addr:192.168.8.98  Bcast:192.168.8.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Figure 4.5: ifconfig of the Wireless Cards in the Client Station [14]

The SoftMAC client station can connect to the dummy internet server through one of the access points. The goal of the test is to see if the implementation of SoftMAC can successfully provide an uninterrupted connection when the client node switches connection between the two access points and still be able to connect to the dummy internet server with an IP address of 192.168.8.97.

In the demonstration, the client station first pings to the dummy internet server through access point 1 using the wireless card with IP address 192.168.8.99. During the pinging, access point 2 is turned on. The MIH function and the link status measurement block will input access point 2 into a table. The next step is bringing access point 1 down. The behavior of the client station is observed. Two scenarios are demonstrated: client station with SoftMAC and client station without SoftMAC.



```
root@localhost:~/script
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

[root@localhost script]# ping 192.168.8.97
PING 192.168.8.97 (192.168.8.97) 56(84) bytes of data.
64 bytes from 192.168.8.97: icmp_seq=1 ttl=64 time=3.67 ms
64 bytes from 192.168.8.97: icmp_seq=2 ttl=64 time=5.06 ms
64 bytes from 192.168.8.97: icmp_seq=3 ttl=64 time=7.55 ms
64 bytes from 192.168.8.97: icmp_seq=4 ttl=64 time=3.64 ms
64 bytes from 192.168.8.97: icmp_seq=5 ttl=64 time=5.02 ms
64 bytes from 192.168.8.97: icmp_seq=6 ttl=64 time=6.13 ms
64 bytes from 192.168.8.97: icmp_seq=7 ttl=64 time=7.58 ms
64 bytes from 192.168.8.97: icmp_seq=8 ttl=64 time=3.50 ms
64 bytes from 192.168.8.97: icmp_seq=9 ttl=64 time=5.57 ms
64 bytes from 192.168.8.97: icmp_seq=10 ttl=64 time=6.14 ms
64 bytes from 192.168.8.97: icmp_seq=10 ttl=64 time=7.53 ms (DUP!)
64 bytes from 192.168.8.97: icmp_seq=11 ttl=64 time=3.63 ms
64 bytes from 192.168.8.97: icmp_seq=12 ttl=64 time=5.01 ms
64 bytes from 192.168.8.97: icmp_seq=13 ttl=64 time=6.12 ms
64 bytes from 192.168.8.97: icmp_seq=14 ttl=64 time=7.51 ms

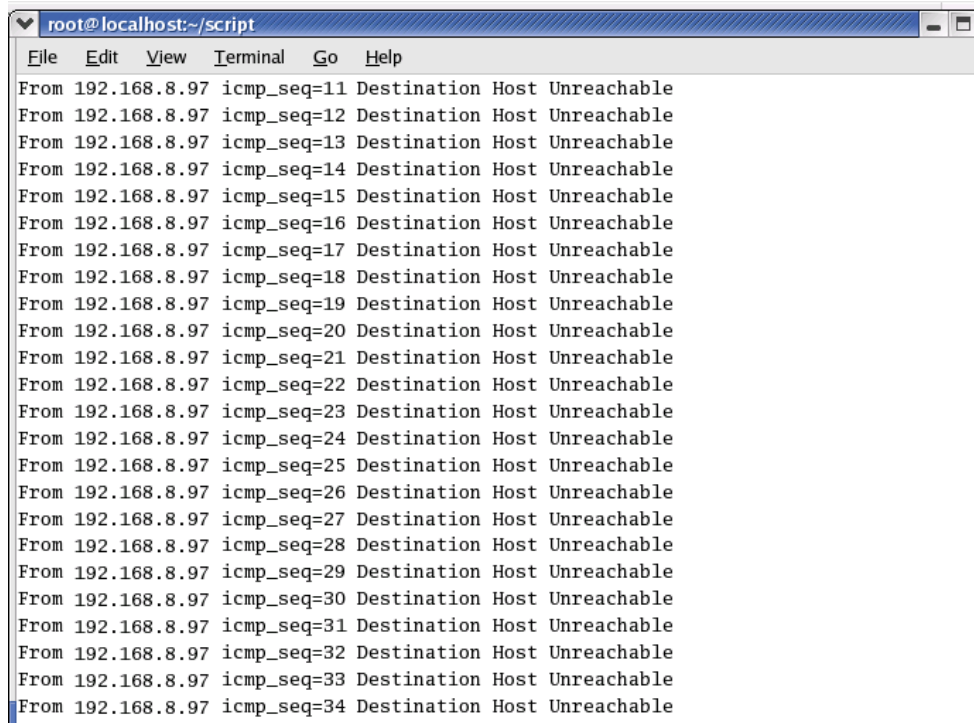
--- 192.168.8.97 ping statistics ---
16 packets transmitted, 16 received, +1 duplicates, 3% packet loss, time 15045ms
rtt min/avg/max/mdev = 3.630/5.551/7.555/1.467 ms
[root@localhost script]#
```

Figure 4.6: Handover Result with SoftMAC [14]

For the client station with SoftMAC, when access point 1 goes down, it will look through the access point table and find the one with the strongest SNR to handover to. In this case, this access point will be access point 2. Figure 4.6 shows that the client is able

to handover to access point 2 within 1 second using the wireless card with IP address 192.168.8.98.

Therefore the client station can have a seamless connection to the dummy internet server without showing any failure (Destination Host Unreachable). This test case is done in the EIT lab office which is under the University of Waterloo Public wireless network; therefore network packet collision exists. From Figure 4.7, it is shown that one duplicated packet (DUP!) occurred because there is one packet loss/collision and the client needs to resend the packet one more time.



```
root@localhost:~/script
File Edit View Terminal Go Help
From 192.168.8.97 icmp_seq=11 Destination Host Unreachable
From 192.168.8.97 icmp_seq=12 Destination Host Unreachable
From 192.168.8.97 icmp_seq=13 Destination Host Unreachable
From 192.168.8.97 icmp_seq=14 Destination Host Unreachable
From 192.168.8.97 icmp_seq=15 Destination Host Unreachable
From 192.168.8.97 icmp_seq=16 Destination Host Unreachable
From 192.168.8.97 icmp_seq=17 Destination Host Unreachable
From 192.168.8.97 icmp_seq=18 Destination Host Unreachable
From 192.168.8.97 icmp_seq=19 Destination Host Unreachable
From 192.168.8.97 icmp_seq=20 Destination Host Unreachable
From 192.168.8.97 icmp_seq=21 Destination Host Unreachable
From 192.168.8.97 icmp_seq=22 Destination Host Unreachable
From 192.168.8.97 icmp_seq=23 Destination Host Unreachable
From 192.168.8.97 icmp_seq=24 Destination Host Unreachable
From 192.168.8.97 icmp_seq=25 Destination Host Unreachable
From 192.168.8.97 icmp_seq=26 Destination Host Unreachable
From 192.168.8.97 icmp_seq=27 Destination Host Unreachable
From 192.168.8.97 icmp_seq=28 Destination Host Unreachable
From 192.168.8.97 icmp_seq=29 Destination Host Unreachable
From 192.168.8.97 icmp_seq=30 Destination Host Unreachable
From 192.168.8.97 icmp_seq=31 Destination Host Unreachable
From 192.168.8.97 icmp_seq=32 Destination Host Unreachable
From 192.168.8.97 icmp_seq=33 Destination Host Unreachable
From 192.168.8.97 icmp_seq=34 Destination Host Unreachable
```

Figure 4.7: Handover without SoftMAC [14]

For the client station without SoftMAC, which is the current technology, Figure 10 shows a disconnected pinging status when access point 1 goes down. When access point 1 goes down, it takes approximately 10 seconds for the client station to reconnect to access point 2 and the dummy internet server using the wireless card with IP address 192.168.8.98. Figure 10 shows the disconnection when it is trying to reconnect through access point 2.

5. Discussion and Future Work

This thesis has investigated the major design requirements for SoftMAC design, and has demonstrated that the implemented prototype has met the design requirements. It proves the possibility and flexibility of using SoftMAC to connect and control Heterogeneous Wireless Network, in order to fulfill seamless handover among multiple heterogeneous wireless interfaces. The thesis has demonstrated that by adding the proposed SoftMAC on top of the traditional MAC layer, the mobile station cannot only perform handover between access points, but also essentially open a door to a wider range of application and services. The data plane of the SoftMAC provides the first priority to the real-time packet to get transmitted and ensure the quality of service.

There are still some limitations of the implementation. One is that the security part of both 802.11 and 802.16 isn't concerned. Another is that only part of the IEEE 802.16 standard is implemented. Due to the popularity of Linux version 2.6, the future design could be making the SoftMAC solution compatible with these two versions of Linux. Furthermore, the SoftMAC interface can be defined and developed to be more functional allowing support for more NICs.

In conclusion, this thesis has provided a kernel-based prototypal solution for wireless extension of SoftMAC on the existing hardware for connecting 802.11 protocol and 802.16 protocol. The implementation has been done directly in kernel space of Linux operating system that manages NIC driver and partially MAC layer. Implementation of proposed approaches was difficult since most of resources are undocumented and vendor specified the IXP425 platform and 802.16 driver. However, the goal of the connection and control of Heterogeneous Wireless Network has been achieved with a known limitation. Experimental result shows that, it is nontrivial to implement SoftMAC approach on existing hardware and standard.

References

- [1] “Wikipedia,” Current Jan. 2008; <http://en.wikipedia.org>
- [2] IEEE 802.16 Documents <http://grouper.ieee.org/groups/802/16/index.html>
- [3] The Linux Networking Architecture: Design and Implementation of Network Protocols in the Linux Kernel, By Klaus Wehrle, Frank Pahlke, Hartmut Ritter, Daniel Müller, Marc Bechler, PRENTICE HALL, 2004
- [4] AiroPeek NX-Wireless Network Protocol Analyzer. <http://www.wildpackets.com/>.
- [5] Ethereal-Network Protocol Analyzer. <http://www.ethereal.com/>.
- [6] MADWiFi. <http://MADWiFi.org/>.
- [7] IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, ISO/IEC 8802-11: 1999(E), Aug. 1999
- [8] A. Rao and I. Stoica, An overlay MAC layer for 802.11 networks, in MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services, (New York, NY, USA), pp. 135–148, ACM Press, 2005.
- [9] M. Neufeld, J. Fifield, C. Doerr, A. Sheth, D. Grunwald, SoftMAC: A flexible wireless research platform, in Proceedings of HotNets IV, 2005.
- [10] Doerr, Weingart, Sicker, and Grunwald, MultiMAC - An Adaptive MAC Framework for Dynamic Radio Networking. In IEEE/ACM DySPAN (Dynamic Spectrum Access Networks), 2005.
- [11] Shugong Xu and Tarek Saadawi, “Does the IEEE 802.11 MAC protocol Work Well in Multihop Wireless Ad Hoc Networks” Challenges in mobile ad hoc networking
- [12] IEEE80211 SoftMAC Layer, <http://SoftMAC.sipsolutions.net/>
- [13] IEEE80211 Subsystem for Linux, <http://ieee80211.sourceforge.net/>
- [14] Jame Ho, Christie Kong, Seamless Wireless Integration, 2008
- [15] H. Javaheri, G. Noubir, Y. Wang (2007). “ Cross-Layer distributed diversity for heterogeneous wireless networks”, in Proceedings of the Fifth International Conference on Wired/Wireless Internet Communication, WWIC'07 , LNCS, Springer-Verlag.

- [16] Haitao Wu, Yunxin Liu, IEEE Globecom 2005, "Layer 2.5 SoftMAC: End-system based Media Streaming Support on Home Networks."
- [17] Haitao Wu, Yunxin Liu, Qian Zhang, Zhi-Li Zhang, SoftMAC: Layer 2.5 Collaborative MAC for Multimedia Support in Multihop Wireless Networks, Mobile Computing, IEEE Transactions, 2007.
- [18] H. Wu, Y. Peng, K. Long, et al., Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement, Proc. IEEE INFOCOM 2002
- [19] K. Langendoen and G. Halkes, Embedded Systems Handbook. CRC Press, Aug. 2005, ch. Energy-Efficient Medium Access Control.
- [20] M. Heusse, F. Rousseau, G. Berger-Sabbatel and A. Duda, "Performance anomaly of 802.11b.
- [21] PAPAGIANNAKI, K., YARVIS, M., AND CONNER, W. S. Experimental characterization of home wireless networks and design implications. Proc. of INFOCOM ,2006.