國立交通大學
電信工程研究所
碩士論文

IEEE 802.11介質存取控制器
設計與實現:
接收部份

Design and Implementation of IEEE 802.11
MAC Controller:
Receiver Part

研究生：洪銘聰
指導教授：陳伯寧 博士

中華民國八十八年五月
IEEE 802.11介質存取控制器設計與實現：
接收部份
Design and Implementation of IEEE 802.11 MAC Controller:
Receiver Part

研究生：洪銘聰
Student：Ming Tsung Hong
指導教授：陳伯寧 博士
Advisor：Dr. Po-Ning Chen

國立交通大學
電信工程研究所
碩士論文

A Thesis
Submitted to Institute of Communication Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University

In Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Electrical Engineering

May 1999
Hsinchu, Taiwan, Republic of China

中華民國八十八年五月
國立交通大學
研究所碩士班

論文口試委員會審定書

本校 __ 電信工程 __ 研究所 ______ 洪 銘 聰 ______ 君

所提論文 (中文) IEEE 802.11 介質存取控制器設計與實現：

接收部份

(英文) Design and Implementation of IEEE 802.11
MAC Controller: Receiver Part

合於碩士資格水準，業經本委員會評審認可。

口試委員： _______________ _______________

_______________ _______________

_______________ _______________

指導教授： _______________ _______________

系主任： 教授

中華民國 88 年 6 月 4 日
授權書
(博碩士論文)

本授權書所授權之論文為本人在國立交通大學電信工程研究所 系統組 八十七學年度第二學期所撰 碩士學位論文。

論文名稱：IEEE 802.11 介質存取控制器設計與實現：接收部份

□同意 □不同意

本人具有著作財產權之論文提要，授予國家圖書館、本校畢業學校及行政院國家科學委員會
科學技術資料中心，得重製成電子資料後收錄於該單位之網路，並與台灣學術網路及科技網
路連線，得不限地域時間與次數，以光碟或紙本重製發行。

□同意 □不同意

本人具有著作財產權之論文全文資料，授予行政院國家科學委員會科學技術資料中心，得不限
地域時間與次數以微縮、光碟重製後發行，並得享該中心微縮小組製作之研究報告、獎勵代表
作、博碩士論文三專資料等值新台幣伍佰元之服務。本論文因涉及專利等智慧財產權之申請，
請將本論文全文延後至民國 ___ 年 ___ 月後再公開。

□同意 □不同意

本人具有著作財產權之論文全文資料，授予教育部指定送繳之圖書館及本校畢業學校圖書館，
為學術研究之目的以各種方法重製，或為上述目的再授權他人以各種方法重製，不限時間與地域，
惟每人以一份為限。

上述授權內容均無須訂立讓與及授權契約書。依本授權之發行權為非專屬性發行權利。依本授權所為之收錄、重製、發行及學術研發利用均為無償。

指導教授姓名：陳伯寧博士

研究生簽名：__________ 學號：8613543
(親筆正楷)

日期：民國 88 年 6 月 4 日

備註：1. 上述同意與不同意之欄若未勾選，本人同意視同授權。
2. 授權第二項者，請再交論文一本予承辦人員。
3. 本授權書已於民國 85 年 4 月 10 日送請審委會修正定稿。
IEEE 802.11 介質存取控制器設計與實現：接收部份

學生：洪銘聰 指導教授：陳伯寧博士

國立交通大學電信工程研究所

中文摘要

「網際網路」，一個蓬勃發展的嶄新領域。「無線通訊」，一個炙手可熱的明星產業。近年來拜可攜式電腦、數位通訊、低功率半導體元件技術的日益精進，網際網路將結合無線通訊的存取技術延伸出一個新的公眾網路型態—無線區域網路。1997 年，IEEE 802.11 Specification 搶先在眾多無線區域網路標準中制訂完成，正式宣告了公眾無線區域網路的來臨。

二年來，雖然 IEEE 802.11 相關無線網路產品不斷問世，但其中介質存取控制層處理器採用的設計方式多為 Embedded System 架構。CPU-based 架構的最大優勢在於可適度修正韌體以達到『彈性化』的設計考量。隨著 IEEE 802.11 Standard 的底定，唯一的彈性優勢也不再是必須的。

為因應成本的效益及高速傳輸的趨勢，論文中提出一套完整的介質存取控制系統方案。我們企圖以硬體方式實現 IEEE 802.11 介質存取控制層的機制，同時提供完整的對主機端與實體層的介面單元與直接記憶體存取控制器模組。最後，此結構將透過電腦輔助設計軟體來實現並將通過層層軟體及硬體的模擬與驗證。
Design and Implementation of IEEE 802.11 MAC Controller: Receiver Part

Student: Ming-Tsung Hong  Advisor: Dr. Po-Ning Chen

Institute of Communication Engineering
National Chiao Tung University

Abstract

「Internet」 has become a new field with full vitality and 「Wireless Communication」 has grown rapidly as a profitable domain in the fashion. Recently, the advance in portable computers, digital communication, and low-power semiconductor technology makes feasible in incorporating Internet and wireless communication access technology and further leads to a new public network configuration—Wireless Local Area Network. In 1997, the standardization of IEEE 802.11 stole messes of wireless LAN specifications’ thunder and thus formally declared the coming of the public wireless local network.

In the past two years, the compliant wireless LAN products come on the scene successively. Most design approaches employ Embedded System architecture in MAC layer controllers. The feasibility of properly revising firmware in the architecture is their best advantage. However, the unique benefit is not necessary anymore after the emergence of the IEEE 802.11 standard.

Under the considerations of cost and high transmission rate, a complete and viable solution for MAC system is presented in the thesis. We attempt to implement the mechanism over IEEE 802.11 MAC layer through hardwire approach and further provide the integrated interface modules for Host end and physical layer device as well as Direct Memory Access controller. The proposed architecture will be realized by CAD design tools, and then simulated and identified through scores of softwares and hardwares.
誌謝

這篇論文得以順利完成，首先要感激我的指導教授陳伯寧博士悉心指導我的論文方向與架構，並讓我由其中瞭解到作研究的方法及應有的精神，奠定我日後人生中更高更穩的基礎。

同時我也十分感謝電子工程所的許騰尹、陳麟旭學長在硬體技術上的支援，並提供我關於如何實現 CRC 的相關文獻，使我的論文進行的更順利。在此更對我的實驗室伙伴們—興明、文碩、鴻賓、秋蔓、金水、嘉峻、韋豪、元豪、清和、立民、維哲、維升、子健—致上謝意，感謝這二年來一同走過的歲月。尤其是我的 partner 富雄，有你的砥礪及互助，才能成就這篇論文。

最後，僅將此論文獻給無時無刻在關懷、支持我的父母。
Contents

中文摘要 ........................................................................................................... I

ABSTRACT ..................................................................................................... II

致谢 .................................................................................................................. III

CONTENTS .................................................................................................... IV

LIST OF TABLES ............................................................................................... IX

LIST OF FIGURES .......................................................................................... 1

CHAPTER 1 INTRODUCTION ........................................................................... 4

1.1 CONTRIBUTIONS .................................................................................... 7

1.2 THESIS OVERVIEW ............................................................................... 8

1.3 LIST OF THE IMPLEMENTED FUNCTIONS .............................................. 9

CHAPTER 2 DESCRIPTION OF STANDARD ............................................. 11

2.1 NETWORK TOPOLOGIES ....................................................................... 11

2.2 GENERAL FRAME STRUCTURE ............................................................... 13

2.2.1 MAC Layer Frame Formats ............................................................... 13

2.2.2 DSSS Physical Layer Frame Formats .............................................. 13

2.3 MAC LAYER FUNCTIONAL DESCRIPTION .......................................... 15

2.3.1 Introduction ..................................................................................... 15

2.3.2 Distributed Coordination Function ................................................. 15

2.3.3 Carrier Sense Mechanism ............................................................... 18

2.3.4 Inter-Frame Space (IFS) ................................................................. 18
CHAPTER 3 SCHEME AND REALIZATION OF MAC SUBLAYER FUNCTION:

RECEPTION PART ........................................................................................................... 22

3.1 OVERVIEW OF THE SYSTEM .............................................................................. 22

3.1.1 Interfacing action ......................................................................................... 22

3.1.2 System Operations in Transmission and Reception ....................................... 27

3.1.3 System Architecture of MAC Sublayer Controller ........................................ 29

BIU module ..................................................................................................................... 30

DMA Control module ..................................................................................................... 30

CSMA/CA module ........................................................................................................... 31

Control Frame Generation module ................................................................................ 31

ABR module ..................................................................................................................... 31

Power Control module .................................................................................................... 31

Selector & TX CRC32 module ....................................................................................... 32

3.2 DETAILS OF CSMA/CA MODULE : RX PART .................................................. 33

3.2.1 Datapath Division ......................................................................................... 33

3.2.2 Control Division ............................................................................................. 34

3.2.3 Reception FSM Division .................................................................................. 34

3.3 CONTROL FRAME GENERATOR ...................................................................... 40

3.3.1 Control Frame Generator Architecture ....................................................... 40

3.3.2 Design of PLCP header and MAC header Generators ............................... 41

3.3.3 Design of Control Finite State Machine ...................................................... 43

3.4 TX/RX CRC DESIGN .......................................................................................... 45

3.4.1 CRC Algorithm and Architecture .............................................................. 45

CHAPTER 4 DIRECT MEMORY ACCESS FUNCTIONS ............................................. 48

4.1 SRAM ARRANGEMENT ..................................................................................... 49

4.2 MEMORY MANAGEMENT .................................................................................. 50

4.3 FLOWCHART OF THE DATA MANAGER ......................................................... 51
CHAPTER 5 VERIFICATION AND SIMULATION RESULTS

5.1 DESIGN FLOW AND TOOLS ................................................................. 52
5.2 GATE LEVEL SIMULATION RESULTS ............................................... 54
  ◊Reception FSM Simulation I................................................................. 54
  ◊Reception FSM Simulation II................................................................. 55
  ◊RX CRC32 Calculation Sub-module .................................................... 56
  ◊PLCP header Generation Sub-module ................................................ 57
  ◊MAC header Generation Sub-module .................................................. 58
  ◊Control FSM Sub-module ................................................................ 59
  ◊Control Frame Generation Module .................................................... 60
  ◊DMA Control Module........................................................................ 61
  ◊ABR Module I.................................................................................... 62
  ◊ABR Module II.................................................................................. 63
5.3 FPGA LEVEL SIMULATION RESULT ............................................... 64
5.4 AREA AND TIMING........................................................................... 65

CHAPTER 6 SUMMARY .............................................................................. 66

6.1 CONCLUSIONS.................................................................................. 66
6.2 FUTURE WORK................................................................................... 67

REFERENCES ............................................................................................. 68

APPENDIX A ............................................................................................... 70

A.1. INTRODUCTION ................................................................................ 71
  A.1.1 SYNOPSIS OF THIS SPEC .......................................................... 71
  A.1.2 DOCUMENT REVISION HISTORY .............................................. 71

A.2. SYSTEM LEVEL DESCRIPTION ...................................................... 72
  A.2.1 FEATURE SET ........................................................................... 72
        A.2.1.1 FEATURES ........................................................................ 72
  A.2.2 SYSTEM LEVEL DESCRIPTION ................................................ 73
A.2.3 SOFTWARE ISSUES ........................................................................................................... 73

A.3. INTERFACING NOTES ........................................................................................................... 75
A.3.1 CSMA/CA AND CONTROL INTERFACE ........................................................................... 75
A.3.2 CSMA/CA AND BASEBAND INTERFACE ....................................................................... 75
A.3.3 BIU MODULE AND HOST SYSTEM INTERFACE ............................................................ 76
A.3.4 SRAM INTERFACE ......................................................................................................... 77
A.3.5 BIU MODULE AND CIS ROM INTERFACE ..................................................................... 77

A.4. THEORY OF OPERATION .................................................................................................... 78
A.4.1 MAC MODULE BLOCK DIAGRAM IN IBSS NETWORK ....................................................... 78
A.4.2 CSMA/CA TX MODULE FIXED REGISTERS ..................................................................... 80
A.4.3 CSMA/CA MODULE STATUS/COMMAND REGISTERS .................................................... 80
   A.4.3.1 Interrupt Tx Status registers ..................................................................................... 80
   A.4.3.2 Interrupt Rx Status registers .................................................................................. 81
   A.4.3.3 Tx Configuration registers ...................................................................................... 81
A.4.4 CSMA/CA RX MODULE FIXED REGISTERS .................................................................... 82
   A.4.4.1 TX Demand Register between RX_FSM and Tx_FSM ............................................... 82
   A.4.4.2 RX_FSM’s Error Status Register .......................................................................... 82
   A.4.4.3 MAC Configure Register ........................................................................................ 83

A.5. UNIT OPERATION ............................................................................................................... 84
A.5.1 CSMA/CA SUBMODULE .................................................................................................. 84
A.5.2 DMA CONTROL MODULE ............................................................................................. 95
A.5.3 BIU CONTROL MODULE ............................................................................................... 97
A.5.4 SRAM MODULE .............................................................................................................. 100
   A.5.4.1 PIN DESCRIPTION ................................................................................................... 100
A.5.5 CONTROL FRAME GENERATION MODULE ................................................................. 102
   A.5.5.1 Interface between control module and baseband module ...................................... 102
   A.5.5.2 Interface between control module and CSMA/CA module .................................... 102
   A.5.5.3 Control Frame module architecture ....................................................................... 103
   A.5.5.4 MAC and PLCP Header generators ....................................................................... 104
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Generating polynomial table</td>
<td>47</td>
</tr>
<tr>
<td>5-1</td>
<td>Attributes of the MAC controller</td>
<td>65</td>
</tr>
<tr>
<td>6-1</td>
<td>Comparison between CPU-based and our proposed architecture</td>
<td>66</td>
</tr>
<tr>
<td>6-2</td>
<td>Table of Future Work</td>
<td>67</td>
</tr>
<tr>
<td>A.1</td>
<td>Address Location for Registers within BIU module</td>
<td>99</td>
</tr>
<tr>
<td>A.2</td>
<td>Read and Write function for I/O and Attribute Memory</td>
<td>99</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1-1  MAC Sublayer Architecture .................................................. 5
Figure 1-2  Harris’s PHY solution .......................................................... 6
Figure 1-3  AMD 79C930’s Architecture .................................................. 7
Figure 2-1  Sketch of an Ad hoc network .................................................. 12
Figure 2-2  Sketch of an infrastructure network ....................................... 12
Figure 2-3  MAC Frame Format ............................................................... 13
Figure 2-4  PLCP Frame Format ............................................................... 14
Figure 2-5  Transmission of MPDU without RTS/CTS .............................. 16
Figure 2-6  Transmission of MPDU using RTS/CTS ............................... 17
Figure 2-7  Hidden Node Effect ............................................................... 17
Figure 2-8  IFS Relationship ................................................................. 19
Figure 2-9  RTS/CTS with Fragmented MSDU ......................................... 20
Figure 2-10 Beacon Transmission in Infrastructure Network ................... 21
Figure 2-11 Beacon Transmission in Independent BSS ......................... 21
Figure 3-1  FPGA Pinout diagram ........................................................... 23
Figure 3-2  Baseband Interface Ports ..................................................... 24
Figure 3-3  Timing diagram of Transmission port .................................. 25
Figure 3-4  Timing diagram of Reception Port ....................................... 27
Figure 3-5  System Architecture of MAC controller .............................. 30
Figure 3-6  Block diagram of CSMA/CA: RX part .................................. 33
Figure 3-7  Flowchart of Reception FSM ............................................... 36
Figure 3-8  Broadcast/Multicast State of Reception FSM ....................... 37
Figure 3-9  Data State of Reception FSM ............................................... 38
Figure 3-10 Control State of Reception FSM ........................................ 38
Figure 3-11 Management State of Reception FSM .................................. 39
Figure 3-12 Control Frame Generator Architecture .............................. 41
Figure 3-13 Control Frame Structure ..................................................... 41
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-14</td>
<td>PLCP header Generator</td>
<td>42</td>
</tr>
<tr>
<td>3-15</td>
<td>MAC header Generator</td>
<td>42</td>
</tr>
<tr>
<td>3-16</td>
<td>Calculation of Duration field</td>
<td>43</td>
</tr>
<tr>
<td>3-17</td>
<td>State diagram of Control FSM</td>
<td>44</td>
</tr>
<tr>
<td>4-1</td>
<td>Block diagram of DMA and ABR modules</td>
<td>48</td>
</tr>
<tr>
<td>4-2</td>
<td>SRAM (32K x8) Allocation</td>
<td>49</td>
</tr>
<tr>
<td>4-3</td>
<td>Pointers of Data Manager</td>
<td>50</td>
</tr>
<tr>
<td>4-4</td>
<td>State diagram of DMA controller</td>
<td>51</td>
</tr>
<tr>
<td>5-1</td>
<td>Design Flow</td>
<td>53</td>
</tr>
<tr>
<td>5-2</td>
<td>RX FSM Simulation: Receiving Probe</td>
<td>54</td>
</tr>
<tr>
<td>5-3</td>
<td>RX FSM Simulation: Receiving Data and Unrecognizable</td>
<td>55</td>
</tr>
<tr>
<td>5-4</td>
<td>RX CRC32 Sub-module: Abort + CRC32 Error</td>
<td>56</td>
</tr>
<tr>
<td>5-5</td>
<td>PLCP header Generator: RTS_2M + ACK_1M</td>
<td>57</td>
</tr>
<tr>
<td>5-6</td>
<td>MAC header Generator: CTS_1M + RTS_1M</td>
<td>58</td>
</tr>
<tr>
<td>5-7</td>
<td>Control FSM Simulation</td>
<td>59</td>
</tr>
<tr>
<td>5-8</td>
<td>Control Frame Generator: Generating RTS_2M</td>
<td>60</td>
</tr>
<tr>
<td>5-9</td>
<td>DMA Controller: Writing operation</td>
<td>61</td>
</tr>
<tr>
<td>5-10</td>
<td>ABR module: Reading process interrupted by Transmission</td>
<td>62</td>
</tr>
<tr>
<td>5-11</td>
<td>ABR Module: Writing process interrupted by Reception</td>
<td>63</td>
</tr>
<tr>
<td>5-12</td>
<td>Reception FIFO simulation at FPGA-level</td>
<td>64</td>
</tr>
<tr>
<td>A.1</td>
<td>MAC Chip Architecture</td>
<td>78</td>
</tr>
<tr>
<td>A.2</td>
<td>CSMA/CA Submodule Architecture</td>
<td>79</td>
</tr>
<tr>
<td>A.3</td>
<td>Data Frame flow chart</td>
<td>85</td>
</tr>
<tr>
<td>A.4</td>
<td>Beacon transmission flow chart</td>
<td>87</td>
</tr>
<tr>
<td>A.5</td>
<td>ATIM transmission flow chart</td>
<td>88</td>
</tr>
<tr>
<td>A.6</td>
<td>RX_FSM</td>
<td>89</td>
</tr>
<tr>
<td>A.7</td>
<td>Data STATE of RX_FSM</td>
<td>90</td>
</tr>
<tr>
<td>A.8</td>
<td>Control STATE of RX_FSM</td>
<td>90</td>
</tr>
<tr>
<td>A.9</td>
<td>Management STATE of RX_FSM</td>
<td>91</td>
</tr>
<tr>
<td>A.10</td>
<td>Broadcast and Multicast STATE of RX_FSM</td>
<td>92</td>
</tr>
<tr>
<td>A.11</td>
<td>TX FIFO Design</td>
<td>93</td>
</tr>
<tr>
<td>A.12</td>
<td>Rx FIFO Unit</td>
<td>94</td>
</tr>
<tr>
<td>A.13</td>
<td>DMA block diagram</td>
<td>95</td>
</tr>
<tr>
<td>A.14</td>
<td>BIU module</td>
<td>97</td>
</tr>
</tbody>
</table>
Figure A.15  Address decode format......................................................... 98
Figure A.16  SRAM (32 K x 8) Partition .................................................... 100
Figure A.17  PLCP Frame Generator .......................................................... 103
Figure A.18  Control Frame Selection....................................................... 104
Chapter 1   Introduction

The advance in digital communications, portable computers, and low-power semiconductor technology results in the rapid growth in wireless Local Area Networks (LANs). Wireless LANs provide not only better portability than wired LANs, but also lower cost in situation where the cable installation is expensive and impractical among manufacturing floors, trade shows, and historical buildings. Moreover, wireless systems have the ability of mobile access to the wired infrastructure.

Although wireless LANs are similar to wired LANs in several respects, many differences exist and lead to additional difficulties in the corresponding design. The key differences are the unreliable nature of wireless communication, the mobility and power saving demands the wireless stations. These differences deeply impact the system design in all layers of networking stack.

One of the main international standards regarding to wireless LANs was recently developed by IEEE, which was identified as IEEE 802.11 [1]. The scope of this standard covers the implementations of the physical layer (PHY) and medium access control (MAC) sublayer. It also standard supports multi-rate data transmission up to 2Mbps.

Functionally, the IEEE 802.11 standard not only provides general contention service for data traffic but also offers contention-free service for real-time traffic. The related MAC architecture is shown in Figure 1-1.
The contention-based fundamental access method adopted in the 802.11 MAC is basically the *Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)*, and is referred to a *Distributed Coordination Function (DCF)*. The DCF is be used over both Ad hoc and infrastructure network topologies. The 802.11 MAC can also incorporate an optional access method called *Point Coordination Function (PCF)*, which is only applicable to infrastructure network configurations. Furthermore, in [2] it is concluded that the centralized mode performs poorly.

At present, there are a lot of compliant products in stock for IEEE 802.11 standard, such as Harris’s PRISM radio chipset for PHY layer and AMD 79C930 for MAC layer, as respectively shown in Figure 1-2 and Figure 1-3. Perhaps, the most common MAC controller for IEEE 802.11 is the AMD 79C930. As shown in the Figure 1-3, its main body is an embedded 80188-microprocessor core. As a result, the MAC solution adopted by AMD 79C930 is not specific for IEEE 802.11 function; and much of the implementation effort of the MAC functionality is deferred to the embedded coding step. In order to promote the chip to WLAN product manufactures, AMD does provide a ready-to-use 80188 embedded code for IEEE 802.11 MAC without any extra charge. Nevertheless, most of the manufactures still develop their own embedded codes for a better fitting to their specific system architectures.

The implementation approach of MAC controller adopted by AMD, although
flexible to standard revision by simply modifying the embedded code, will unavoidable limit the system performance by the processing speed of the 80188 core. In addition, its cost, if compared to a specially-designed ASIC, remains high, which results in a high cost of WLAN products. In order to promote the WLAN market that is currently stuck due to its cost, a low cost design seems necessary. This motivates our project that focuses on a specially designed ASIC of IEEE 802.11 MAC controller.

Figure 1-2  Harris’s PHY solution
1.1 Contributions

In this thesis, we investigate the design issues of implementing IEEE 802.11 via VLSI approach. Our contributions are briefly listed below:

1. Using the **System-On-Chip (SOC)** approach to implement the DCF service.
2. Providing a feasible hardware architecture that fits the high-speed transmission tendency.
3. Making the programming of host driver almost effortless due to a built-in control frame unit.
4. Implementing three **parallel** Cyclic Redundancy Check (CRC) units, which is based on Charles Zukowski's architecture [5].
5. Designing a specific built-in Direct Memory Access (DMA) module for a
fats data exchange between host and adapter enforced with our MAC controller.

1.2 Thesis Overview

This thesis is organized as follows:

Chapter 2 describes concisely the basic medium access protocol, i.e., DCF, which is the kernel of the standard.

Chapter 3 describes in detail the architectures and algorithms of our core design, as well as the encountered obstacles. Essentially, our MAC controller consists of several modules. Some of them, such as the control frame generating module and part of, the CSMA/CA module, will be presented in the thesis. The remaining modules, such as bus interface unit (BIU) and configuration registers, are covered in [3].

Chapter 4 presents the extra effort involving DMA module and ABR module. Although these modules are independent of MAC sublayer protocol, their functions are needed for the data exchange between memory and FIFO.

Chapter 5 displays the simulation results. These simulations validate the feasibility of our approach.

In Chapter 6, we summarize the features of our design and address some possible future work along with the bottlenecks at the MAC level of the coming high-speed version of IEEE 802.11 standard.

A design specification (datasheet) of our MAC controller is attached as Appendix. It consists of a complete functional description of each module.
1.3 List of the Implemented Functions

- **Distributed Coordination Function**
  - Net Allocation Vector (NAV) Function
  - Inter-frame Space Usage & Timing
  - Random Backoff Function
  - DCF Access Procedure
  - Random Backoff Procedure
  - Recovery Procedures & Retransmit Limits
  - RTS/CTS Procedure
  - Directed MPDU Transfer
  - Broadcast & Multicast MPDU Transfer
  - MAC level Acknowledgement
  - Duplicate Detection and Recovery

- **Multi-rate Data Transmission Support**
  - 1Mbps
  - 2Mbps

- **Timing Synchronization**
  - Timing in an Independent BSS
  - Beacon Generation Function
  - TSF Synchronization & Accuracy
  - IBSS Initialization
  - Passive Scanning

- **IBSS Power Management**
  - Initialization of Power Management
  - STA Power State Transition
• ATIM and Frame Transmission

**Frames directly Processed by MAC chip**
• RTS
• CTS
• ACK

**Frame Exchange Sequences**
• Basic Frame sequences

**MAC Addressing Functions**
• STA Universal Individual IEEE 802 Address
• BSS Identifier Generation
• Receive address Matching

**DSSS Physical Layer Functions**
• PLCP Sublayer Procedures
• Preamble prepend on TX
• PLCP frame format

**Extra Effort**
• Bus Interface Unit
• Direct Memory Access Controller
Chapter 2  Description of Standard

2.1 Network Topologies

The Basic Service Set (BSS) is the fundamental building block of the IEEE 802.11 wireless LAN architecture. A BSS is defined as a group of stations that are under the direct control of a single coordination function, i.e., a DCF or PCF which is defined in section 2.3.2 below.

A single BSS can be used to form an Ad hoc network. An Ad hoc network is a deliberate grouping of stations into a single BSS for the purpose of internetworked communications without the aid of an infrastructure network. Figure 2-1 is an illustration of an Ad hoc network. The formal name of an Ad hoc network in the IEEE 802.11 standard is an **Independent Basic Service Set (IBSS)**. Any station can establish a direct communication session with any other station in the same IBSS, without the requirement of tunneling traffic through a centralized access point (AP).

\[\text{1}\]

In contrast to the Ad hoc network, infrastructure networks in the context of IEEE 802.11 are established with an Access Point (AP). The AP is analogous to the base station in a cellular communication network. The AP supports range extension by providing the necessary integration points for network connectivity between multiple BSSs, thus forming an Extension Service Set (ESS). The ESS consists of multiple BSSs that are integrated together using a common Distributed System (DS). The DS can be considered as a backbone network that is responsible for MAC level transport of MAC Service Data Unit (MSDU). Figure 2-2 illustrates a simple ESS developed with two BSSs, and a DS.

---

\[\text{1}\] We only implement the functions required by Ad hoc network.
Figure 2-1  Sketch of an Ad hoc network

Figure 2-2  Sketch of an infrastructure network
2.2 General Frame Structure

2.2.1 MAC Layer Frame Formats

Each MPDU shall comprise the following basic components:

(a) A **MAC header**, which comprises frame control, duration, address, and sequence control information.

(b) A variable length **Frame Body**

(c) An IEEE standard **CRC32**.

Figure 2-3 depicts the general MAC frame format. Therefor, the MAC header format varies with **Frame Type**.

![802.11 MAC Header](image)

Figure 2-3  MAC Frame Format

2.2.2 DSSS Physical Layer Frame Formats

Figure 2-4 shows the format for the **Physical Protocol Data Unit (PPDU)** including the PLCP preamble, the PLCP header and the MPDU. The entire PLCP preamble and header shall be transmitted using 1Mbps/DBPSK modulation. Each of the fields are described in detail below.
Figure 2-4 PLCP Frame Format

◆ SYNC & SFD
These two fields are formed by 144-bit constant pattern.

◆ SIGNAL Field
The 8-bit signal field indicates to the PHY the modulation that shall be used for transmission and reception of the MPDU.

◆ LENGTH Field
This field indicates the number of microseconds required to transmit the MPDU.
2.3 MAC Layer Functional Description

2.3.1 Introduction

The MAC sublayer is responsible for the channel allocation procedures, protocol data unit (PDU) addressing, frame formatting, error checking and fragmentation and reassembly. IEEE 802.11 supports three different type of frames: Management, Control, Data. The management frames are used for station association and disassociation with the AP, timing and synchronization, and authentication and deauthentication. Control frames are used for handshaking during the contention period (CP), for positive acknowledgements during the CP, and to end the CFP. Data frames are used for the transmission of data during the CP and CFP, and can be combined with polling and acknowledgements during the CFP.

2.3.2 Distributed Coordination Function

The DCF is the fundamental access method used to support asynchronous data transfer on a best effort basis. As identified in the specification, all stations must support the DCF. The DCF is based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol, with rotating backoff windows and optional RTS/CTS message exchange to prevent from hidden terminal effect. The traditional collision detection technology, such as CSMA/CD, was not adopted since the wireless stations are unable to listen to the channel for collisions while transmitting. So the CSMA/CA protocol is employed instead to reduce the collision probability among multiple stations sharing a common medium, at the point where collision would most likely occur. Immediately after the medium becomes free following a busy medium (as indicated by CS function) is when the highest probability of a collision occurs. This is because multiple stations could have been waiting for the medium to become available again. This is why a random backoff arrangement is needed to resolve medium contention conflicts.
More specifically, a station that intends to transmit and sense the channel busy will wait for the end of the ongoing transmission. After the channel clear, the station will wait for an additional time period of DIFS (Distributed Coordination Inter-Frame Space); and then randomly selects a number within the backoff window. The number will decrease by one after each time slot period of channel clear. If no other station starts transmitting before the number goes to zero, it starts to transmit; otherwise, it will freeze its back-off counter, wait for the end of this transmission, and now only wait for the remaining number to decrease to zero under channel-clear count-down. Figure 2-5 is an illustration of transmission of MPDU.

The above access mechanism can optionally be extended by the RTS/CTS message exchange. Before the transmission of a data frame, a short control frame named RTS (Request To Send) is sent to the receiving station. A CTS (Clear to Send) control frame to indicate that the receiving station is ready to take the data frame. Both the RTS and CTS control frames contain the Duration information (in microseconds) of how long the channel will be used, including the inter-frame spaces and the time to obtain the returned ACK message for the MPDU as shown in the Figure 2-6. The RTS/CTS sequence exchange can also resolve the Hidden node effect, which is illustrated in Figure 2-7. When data transfer is proceeding in the medium, other transmitters contending for the medium may not hear the transmitting station. And yet these transmitter may receive CTS frame issued from the receiving station. Consequently, RTS/CTS exchange can fairly depress the hidden node effect.

![Figure 2-5](image_url)
Figure 2-6  Transmission of MPDU using RTS/CTS

Figure 2-7  Hidden Node Effect
2.3.3 Carrier Sense Mechanism

In IEEE 802.11, carrier sensing is performed both at the air interface, referred to as physical carrier sensing, and at the MAC sublayer, referred to as virtual carrier sensing. Physical carrier sensing detects the presence of the IEEE 802.11 WLAN users by analyzing the activity of the channel via relative signal strength on the air.

Virtual carrier sensing is used by a source station to inform all other stations how long the channel will be utilized for a complete transmission sequence of a MAC protocol data unit (MPDU). The source station sets the duration field in the MAC header of Request-to-Send (RTS) and data frame; likewise, the destination station sets the duration field in the MAC header of Clear-to-Send (CTS) control frame. The duration field indicates the amount of remaining time (in microsecond), which the channel will be utilized to complete the exchange of the frames in a transmission sequence after the end of present frame. Stations detecting a duration field in a transmitted MPDU should adjust their Network Allocation Vector (NAV), which indicates the amount of time that must elapse until the current transmission session finishes. Therefore, the channel is marked busy if either the physical or virtual carrier sensing mechanism indicates a busy media.

2.3.4 Inter-Frame Space (IFS)

The time interval between two consecutive frames is called the inter-frame space. Priority access to the wireless medium is controlled through the use of IFS time intervals between the transmission of frames. The IFS intervals are mandatory periods of idle time on the transmission medium. Four IFS intervals are specified in the standard:

(a) SIFS Short Inter-Frame Space
(b) PIFS Point coordination function Inter-Frame Space
(c) DIFS Distributed coordination function Inter-Frame Space
(d) EIFS Extended Inter-Frame Space

The SIFS interval is the smallest IFS, followed by PIFS, DIFS and EIFS. Stations required to wait for a SIFS have higher access priority over those stations required to
wait for a PIFS or DIFS before transmitting.

Figure 2-8  IFS Relationship

2.3.5 Updating the Network Allocation Vector (NAV)

Stations receiving a valid frame shall reset their NAV counter with the Duration field, if the new NAV value is greater than the current NAV one, and the frame is not addressing to the receiving station. Figure 2-6 displays the setting of NAV counter for stations that receive the RTS, while other stations may only receive the CTS, resulting in a shorter NAV bar. Figure 2-9 describes a more complicated NAV setting using RTS/CTS with fragmented MSDU. Each frame contains information that defines the duration of the next transmission. Both Fragment 0 and ACK 0 shall contain duration information for NAV updating to indicate a busy channel until the end of ACK 1. This shall continue until the last fragment (Fragment 1), which shall contain a duration field of one ACK time plus one SIFS. The last frame of the session is ACK 1, which shall have the duration set to zero. Accordingly, every Fragment and ACK act as a dummy RTS/CTS exchange.
2.3.6 Synchronization Function

Because the power saving capability is necessary for wireless stations, to keep from unnecessary power consumption is essential. This can be attained by powering up the stations only when a received frame is arrived. In order to achieve this objective, all stations shall maintain a local synchronous timer called **Timing Synchronization Function (TSF)** timer, which synchronized to a common clock. The TSF maintenance algorithm used in the infrastructure network differs from that adopted in the Ad hoc topology.

In the infrastructure network, the AP shall be the timing master. In order for the other stations to synchronize with the timing master, the AP shall periodically generate and transmit a Beacons frame that contains a copy of its TSF timer. Other stations shall always adjust their timers according to the TSF information in Beacons sent from AP. Figure 2-10 illustrates the general situation of beacon transmission in the infrastructure network.
Figure 2-10  Beacon Transmission in Infrastructure Network

In contrast, the Timing Synchronization Function in an IBSS is performed via a distributed algorithm. All members in the BSS shall participate in Beacon generation and transmission. At each Target Beacon Transmission Time (TBTT), every station shall calculate a random delay; and if no Beacon has arrived during the period of the delay, then it sends a Beacon at the end of this period. This is described in Figure 2-11.

Figure 2-11  Beacon Transmission in Independent BSS
Chapter 3
Scheme and Realization of MAC Sublayer Function: Reception Part

3.1 Overview of the System

3.1.1 Interfacing action

The Figure 3-1 is the connection diagram of our FPGA sample. It consists of three primary digital interfaces: PCMCIA interface, External memory interface, and Baseband interface.
To cope with the portability of WLAN products, the Bus Interface Unit (BIU) between Host and MAC controller implemented is chosen to be the Personal Computer Memory Card International Association (PCMCIA) revision 2.0, with support for Plug & Play (PNP). This interface is used to initialize the MAC controller through slave I/O accesses. Device operation can also be monitored through the interface by accessing proper MAC controller’s registers. Network data is transferred to/from the software driver through memory accesses. The PCMCIA interface supports an 8-bit wide data bus, and the address line is 15-bit in length.

Our MAC controller supports the Card Information Structure (CIS) and Card Configuration Registers (CCR) defined in the PCMCIA v2.0 standard. The registers are located at an external read-only device, such as EPROM. Through these registers, initialization of the system can be done automatically. The detailed functional description of BIU could be found in [3].

Network data is stored in a 32Kx8 SRAM. Therefore, an external memory bus interface is required in the MAC controller, which is used by the MAC controller to gain access to SRAM for fetching and storing network data. The data stored in the SRAM comes from two separate sources: transmitted data from Host system, and received data from Baseband processor. The transmitted data from Host system are passed to the MAC controller through the system bus interface, and will be automatically rerouted to the memory bus interface to reach the SRAM. Similarly, via the memory bus interface, the MAC controller accesses the data to/from the SRAM for RX/TX actions. The memory bus interface is controlled by a specific DMA unit, which will be described in detailed in Chapter 4.

The Baseband interface acts as an interface between MAC controller and Baseband processor. The operation mode of Baseband interface adopts Media Independent Interface (MII) standard, which is the same as typical Ethernet interface. The Baseband interface can be divided into three categories: Control port, Transmission port, and Reception port (cf. Figure 3-2).
These ports are:

- **Control Port**, which is used to retrieve control information from the physical layer, which is extracted by the Baseband processor from received frames.

- **Transmission Port**, which is used to output the data that needs to be transmitted to Baseband processor.

- **Reception Port**, which is used to accept the received demodulated data from the Baseband processor.

**Control Port**

The parallel control port is used to understand the physical layer header message of received frames and channel status. The **Signal** and **Length** are two fields of PLCP (Physical Layer Convergence Protocol) header, which are striped by Baseband processor. Length field provides the time required to complete the reception of the
received frame. Signal field indicates the data rate of the received frame. The MAC controller then processes the demodulated frames based on the information of these two fields. CCA (Clear Channel Assessment) provided by Baseband processor informs the MAC processor the status, i.e., busy or idle, of the air. The MAC controller also monitors the CRC_16_err signal to assure the correctness of PLCP header. An external device, either the Baseband processor or a standalone oscillator, shall provide the MAC controller with the 44MHz system clock via Sys clk pin.

Adoption of parallel MII interface fetches more data at once and hence, speed up the attainable data rate. It is also convenient for the integration of the MAC controller with the Baseband processor into single chip.

◆ Transmission Port

The Tx_Data data bus is 4-bit in width, according to the MII specification. The parallel data is outputted to the Baseband processor through TX_Data at every rising edge of TX_CLK provided by Baseband processor. Its timing scenario is shown in Figure 3-3.

![Timing diagram of Transmission port](image)

Figure 3-3 Timing diagram of Transmission port

The Transmission Finite State Machine (TX FSM) inside the MAC controller initiates the transmission sequence by asserting Tx_ready. Tx_ready reminds the Baseband processor that there exists pending frames to be transmitted. The Baseband processor responds by first activating 1Mbps (250KHz x 4) TX_CLK to MAC
controller. At every TX_CLK rising edge, data is sent to the Baseband processor via Tx_Data in sequence. The process continues until Tx_ready goes back to its inactive state, indicating the end of the frame is reached.

When the Baseband processor, e.g., Harris HFA3824, is configured as a Preamble and PLCP header generator (in addition to its normal Baseband functionality), discontinuity at the junction of PLCP header transmission and MPDU transmission may constantly happen. Thus, in our design, the generation of Preamble and PLCP header is no longer a task of Baseband processor. Instead, all the fields (including Preamble, PLCP header and MPDU) of Management frames and Data frames are prepared by the host software driver. Nevertheless, due the urgentness of Control frames, the Control frames are generated internally by hardware-based Control Frame Generation Module within MAC controller.

According to IEEE 802.11 standard, the Preamble and PLCP header parts must be transmitted by 1Mbps rate. The remaining part, i.e. MPDU, can optionally be transmitted at a higher data rate. TX_CLK can be raised up to 500KHz (which corresponds to 2Mbps data rate). It is illustrated in Figure 3-3.

❖Reception Port

The timing diagram in Figure 3-4 illustrates the relation among various signals of Reception Port (RX port). Similar to TX port, the Rx-Data bus is 4-bit in width. The received data port inputs the demodulated data from Rx_Data in parallel at every rising edges of RX_CLK. Rx_ready must be set throughout the reception operation.
The Rx_ready, an output from Baseband, is used to notify the MAC controller that there is a receiving frame, and it shall be active after the confirmation of CRC16. The Baseband then outputs RX_CLK and MPDU to MAC controller until Rx_ready goes inactive.

In contrast to transmission operation, the Preamble and PLCP header of the received frame are processed by Baseband processor, and will not be passed to the MAC controller. So the frequency of RX_CLK based on the received Signal field is fixed throughout the whole reception operation. Note that even if the CRC16 checks failed, the receiving operation could still go on. The MAC controller can either proceed to continue the reception operation or take other action when an CRC16 error is occurred.

3.1.1 System Operations in Transmission and Reception

Transmission Sequence of Data and Management Frames

The data path of frame transmission, from host interface, through MAC controller, down to Baseband processor, is described as follow:
a. **Software driver** initializes the MAC controller.

b. The Software Driver first places either Data frames or Management frames desired to transmit into a predefined TX buffer area over the SRAM through PCMCIA interface of the MAC controller.

c. The TX operation is then initiated by a **Tx_request** command issued by software driver.

d. The first operation for Transmission FSM in response to the TX_request command is to load the first 48 bytes of the PPDU into TX FIFO.

< **CSMA/CA** >

e. When the Transmission FSM observes **CCA** (IDLE) for a specified DIFS time, followed by the expiration of a pending backoff time, the Transmission FSM initiates TX operation.

< **Power Ramp-up** >

f. Power up the TX sections of IF, RF, and Baseband chipset.

g. Provide data through Tx_Data pins that comes from TX FIFO after the assertion of **Tx_ready**.

h. The Baseband processor shall locate the **Signal field** of PLCP header to determine the data rate required for the MAC portion of the frame.

< **Note** >

The Preamble and PLCP header shall be transmitted at 1Mbps, while the MPDU may be transmitted at either 1Mbps or 2Mbps.

i. Upon the end of the frame transmission operation, power off the Baseband, IF, RF, etc.

j. End of transmission.

◆ **Reception Sequence of Data and Management Frames**

a. MAC controller enables the Reception FSM whenever **Rx_ready** goes active or software driver dictates **RX_request** command

b. The Baseband processor strips the Preamble and PLCP header, and also checks the CRC16. Baseband then synchronously sends the MPDU data to the RX FIFO of the MAC controller.

c. As new RX data continues to arrive, the previous content of the RX FIFO will be moved to SRAM through **local DMA**. Meanwhile, some
fields of the *MAC header*, such as Frame Control, Duration field etc., will be parsed. If CRC16 is corrupted, the RX_FSM will block the reception of the frame.

d. If *More Fragment* bit equals *zero*, Reception FSM will ramp-down the physical layer components after receiving the last byte of the frame; otherwise, Reception FSM will return to *Wait_Rx_ready* State to wait for next fragment.

e. MAC controller *interrupts* the Host system, indicating there are outstanding frames in the SRAM.

f. End of reception process.

### 3.1.2 System Architecture of MAC Sublayer Controller

Figure 3-5 illustrates the relationship among various devices. It also shows the primary modules of MAC controller. In the Figure, the modules drawn in terms of thick solid lines will be described in detail in the latter sections. The modules drawn by thin solid lines can be found in [3]. The only one module surrounded by dashed lines is deferred to the future work. The functionality of each block is briefly described as follow:
BIU module

This module follows the PCMCIA standard, ver 2.0. The BIU module interfaces the Host system and the MAC controller. The initialization of the MAC controller is also done through this module.

DMA Control module

DMA controller handles the frame transfer among devices, such as Host, SRAM, and FIFO. The SRAM is designed to be 32K x 8. In order to work efficiently, the DMA controller allows two kinds of operations--- data transfer between host and SRAM, and data transfer between SRAM and FIFO to work concurrently in an interleaved fashion. The detail is placed in Chapter 4.
CSMA/CA module

The CSMA/CA module is the core of the MAC controller. It performs the IEEE 802.11 DCF mechanism. It consists of a Reception part and a Transmission part. This thesis only focuses on the Reception part. The transmission part is described in another thesis [3].

Control Frame Generation module

In addition to the CSMA/CA module, another special characteristic of our architecture is the implementation of Control Frame Generator. It generates three kinds of control frames, i.e. RTS, CTS and ACK, independently to reduce the MAC processing time. In order to match the other two types of frames, Data and Management frames, the control frames generated by Control Frame Generator contain not only the MAC body, but also the Preamble and PLCP header.

ABR module

The content of the SRAM can come from two sets of data buses with separate address lines. One of them is from the Bus Interface Unit. The other springs from TX FIFO and RX FIFO. Thus, ABR is used to guarantee that only one of above two sets is active.

Power Control module

The MAC controller shall be able to control the power of all external devices, such as Baseband processor, IF/RF chipset, synthesizer, etc. The functions of this module break into two portions: power management function and power sequencing function. The power management function will enable some essential sections of chipset, and disable the other unnecessary sections according to different operations. The power sequencing function provides an algorithm of transceiver power ramp up/down in accordance with the time required to be warmed up. For instance, when transmission actions operate, MAC controller can power up the Antenna first, and then enable synthesizer, IF/RF, and Baseband processor in sequence. This procedure can protect from additional noise generated by sudden impulses.
In our current design, only one power_en pin is implemented. A more thorough implementation is deferred to future work.

**Selector & TX CRC32 module**

This module consists of a Multiplexer (MUX) and a CRC32 generator. The MUX is used to select the valid data stream from either Data/Management frame path or Control frame path. No matter what kind of frames is transmitted, the CRC32 field will be appended, immediately following the payload.
3.2 Details of CSMA/CA module: RX part

The related functional blocks of Reception part of CSMA/CA module are shown in Figure 3-6. It could be divided into three partitions: Reception FSM Division, Control Division, and Datapath Division.

![Figure 3-6 Block diagram of CSMA/CA: RX part](image)

3.2.1 Datapath Division

The datapath division consists of RX FIFO, RX FIFO controller, and Grabbed registers. RX FIFO holds 1 byte of data. The RX_CLK from MII interface to RX FIFO is either 250 kHz or 500 kHz. But, the clock rate for the data stream from RX FIFO to DMA controller is up to 44 MHz (system clock).

RX FIFO controller controls the overall operations of datapath division. It is activated by Rx_ready signal. When the frame continues to arrive, RX FIFO
controller will move the first 24 bytes (including Frame Control field and Sequence Control field) of the frame into the Grabbed register so that necessary information can be provided to Reception FSM and control frame generation module. Once the Grabbed register fetches the information that Reception FSM needs, the FSM_en signal will be set in order to initiate the Reception FSM. Meanwhile, RX FIFO controller will filter out the Control frames, and only Data/Management frames to enter the SRAM.

3.2.2 Control Division

The control division includes Length counter, NAV timer, and RX CRC32 detector. The NAV timer performs virtual carrier sensing mechanism by counting the reserved period of the channel. The countdown number of NAV timer is obtained from Duration field of the received frames, which is stored in the grabbed register.

The Length counter is used to count the number of received bits according to Length information provided by Baseband processor. When the counter reaches the limit, it will notify Reception FSM to finish the state transition.

The RX CRC32 checker has the same hardware structure as TX CRC32 generator. It is responsible for checking the integrity of the received MPDU. In our design, RX CRC32 adopts a 4-bit-input parallel implementation so that it can completely synchronizes with the RX FIFO and 4-bit wide MII interface. A CRC_32_err will inform the Reception FSM to discard the corrupted frame.

3.2.3 Reception FSM Division

The Reception FSM is the heart of the CSMA/CA module. Reception FSM processes the received frame based on the information in the grabbed register and control division. During the working period of the Reception FSM proceeds, all the intermediate decisions will be placed at the Rx Part Interface (RPI) register file. The RPI register file is constituted with three kinds of registers: TX Demand Register, Error Status Register, and Control Register. TX Demand register records the types of
frames received, such as ACK, CTS, ATIM and Beacon, and passes the information to the Transmission FSM. The Error Status register marks the types of errors occurred. The Control register records the information that control frame generator demands. The explicit definition of RPI register file can be found in Appendix A [cf. A3.1, A4.4.1, and A4.4.2].

The complete flowcharts of Reception FSM are illustrated in Figure 3-7. The detailed statements within Broadcast/Multicast State, Data State, Control State, and Management State are also represented respectively in Figures 3-8, 3-9, 3-10, and 3-11.

As shown in Figure 3-8, Reception FSM is initiated by either Rx_request command issued from software driver, or FSM_en signal sent from grabbed register. After initiation, Reception FSM will leave the Idle State, and enter the Inspect State. The primary statements of Inspect State are the examinations of the CRC16, version bits, and Address 1 field (Destination Address). Checking Address 1 field distinguish among Broadcast/Multicast frames and Unicast frames, as well as “For-me” and “Not-for-me”. If Unicast frame is received, one of the three states will be entered, according to its *Type* field, which are Data State, Management State, and Control State.
Figure 3-7  Flowchart of Reception FSM

If any incorrectness occurs during the reception process, it will transit to Abort State to discard the current received frame. The Abort State is basically to reset all the related functional blocks. More specifically, Abort State will first inform the RX FIFO controller and DMA controller to throw away the current received data in the RX FIFO and SRAM. Then, it will signify to the RX FIFO controller to block out the next incoming data. Also done in the Abort State is the reset of the Length counter and the RX CRC32 detector, and the refreshes of the RPI register file. After finishing all the above tasks, the FSM will return to the Idle State to wait for next frame.

If any of the following four types of frames--- Data type frame, RTS frame,
ATIM frame and Probe Response--- is received, a En_Rx signal will be launched to enable the Control Frame Generator for the transmission of ACK/CTS frame. Note that at this moment, the Address 2 field (Source Address) will be used to distinguish multicast/broadcast frames transmitted by itself in the BC/MC State. As described in the IEEE 802.11 standard, multicast/broadcast frames need no acknowledgement. Hence, the control frame generator will never be enabled at BC/MC State.

![Diagram](image)

**Figure 3-8** Broadcast/Multicast State of Reception FSM
Figure 3-9  Data State of Reception FSM

Figure 3-10  Control State of Reception FSM
Figure 3-11  Management State of Reception FSM
3.3 Control Frame Generator

In our original plan, both the Management frames and Control frames shall be generated and also processed by the MAC controller. Host only needs to handle the Data Frame. However, considering that a full implementation of these may take a longer time than this project allowed (the first term of this project ends at the end of June of 1999), only Control frames are taken care by individual hardware circuit. As a consequence, the host driver needs to handle not only Data frames, but also the Management frames at the current stage of the implementation. In our current design, Control Frame Generator can generate RTS, CTS and ACK within the time interval that is much shorter than SIFS (10us). It needs to point out that there are actually six types of control frames. Lack of PS-Poll, CF-END and CF-END+CF-ACK results in that this MAC controller can only supports DCF service over an Ad hoc network.

3.3.1 Control Frame Generator Architecture

Figure 3-12 illustrates the architecture of control frame generation module. It consists of several sub-field generators and a Control FSM. The simplest generator is the Preamble generator, since the content of the Preamble is fixed.
3.3.2 Design of PLCP header and MAC header Generators

As shown in Figure 3-13, the fields containing in the MAC header of control frames may differ for different subtype. For example, only RTS frame needs Address 2 field (Source Address). The designs of PLCP header and MAC header generators are displayed in Figures 3-14 and 3-15.

Figure 3-12 Control Frame Generator Architecture

Figure 3-13 Control Frame Structure
Figure 3-14  PLCP header Generator

Figure 3-15  MAC header Generator
A special consideration is the calculation of the Duration field, which are formulated below:

**RTS’s Duration value** = \( \text{Data} + \text{CTS} + \text{ACK} + 3 \times \text{SIFS} \)

**CTS’s Duration value** = Duration (received RTS) – CTS – SIFS

**ACK’s Duration value** differs from *More Fragment* bit in received Data

\[
\begin{align*}
\text{Duration value} &= 0, & \text{if More Fragment bit equals zero;} \\
\text{Duration value} &= \text{Duration (received Data)} - \text{ACK} - \text{SIFS} & \text{o.w.}
\end{align*}
\]

These formulas are clearly depicted in Figure 3-16.

![Diagram](image)

**Figure 3-16  Calculation of Duration field**

### 3.3.3 Design of Control Finite State Machine

As mentioned earlier, several generation blocks jointly produce all the required fields of the control frame. Therefore, a Control FSM is needed to multiplex the output of these generation blocks into the MII interface. Figure 3-17 shows the state diagram of the Control FSM. The Control FSM will be initiated upon the reception of the request signals, which is generated from either TX FSM (En_Tx) or RX FSM (En_Rx). As anticipated, Control FSM will first transmit the Preamble field, followed by the PLCP header and MAC header.
Figure 3-17  State diagram of Control FSM
3.4 TX/RX CRC Design

In our system architecture, there are three CRC hardware blocks: CRC16 generator (inside of Control Frame Generation Module), TX CRC32 generator (inside of SEL/CRC32 Module), and RX CRC32 detector (inside of CSMA/CA Module). CRC16 generator and TX CRC32 generator are used to produce the CRC fields attached immediately behind PLCP header and MAC header. The RX CRC32 detector is responsible for the verification the integrity of the received MPDU. All these three CRC hardware blocks are implemented based on similar architecture.

A conventional serial CRC architecture is depicted in the Figure 3-18. It suffers from the limitation that the information stream must be bit-serial. However, our MII interfacing scheme is a parallel 4-bit data bus. In order to cope with the MII interface, also to reduce the S/P (Serial-to-Parallel) and P/S (Parallel-to-Serial) efforts, we adopt a parallel CRC architecture proposed in [5]. The following sections discuss their details explicitly.

3.4.1 CRC Algorithm and Architecture

Let \( X(t) = [x_0 \ x_1 \ \ldots \ x_{t-1}] \) be the remainder after \( t \) conditional subtractions, the state transition equation for circuit using the serial architecture of Figure 3-18 can be written as (where \( \oplus \) represents modulo-2 addition):

\[
X(t) = [x_0 \ x_1 \ \ldots \ x_{t-1}] \oplus [0 \ | \ I_{t-1}] \oplus ([x_{t-1}] \oplus z) \cdot G
\]

where \( G = [g_0 \ g_1 \ \ldots \ g_m] \) and, \( z_t \) is the input bit at time \( t \).

In [5], the authors found that the shift and conditional subtract operations can be merged together within a single clock cycle. In other words, if \( k \) of the total \( m \) subtractions are combined, \( k \) message bits must arrive during each clock cycle and only \( m/k \) clock cycles are required to compute the remainder. Specifically, let \( Z(t) = [z_t \ z_{t+1} \ \ldots \ z_{t+k-1}] \) be a vector containing a parallel group of \( k \) message bits as in [6].
By combining \( k \) conditional subtractions, the state transition equation for the parallel CRC circuit becomes

\[
X(t + k) = [x_0x_1...x_n-k-1][0 \mid I_n-k] \oplus ([x_n-kx_{n-k+1}...x_{n-1}] \oplus Z(t)) D
\]

where \( T^j \) is the \( j \)th power of the matrix \( T \)

\[
D = \begin{bmatrix}
G \\
GT_1 \\
. \\
. \\
. \\
GT_{k-1}
\end{bmatrix}
\]

\[
T = \begin{bmatrix}
0 & 1 & 0 & . & 0 \\
0 & 0 & 1 & . & 0 \\
. & . & . & . & 1 \\
g_0 & g_1 & g_2 & . & g_{n-1}
\end{bmatrix}
\]

In TX/RX CRC32 submodules, we build a CRC generator using \( k=4 \) and \( n=32 \). The IEEE standard generating polynomial \( g_r(x) \) of degree 32 is listed in the table 3-1. The equations that describe the state transition for this circuit now becomes

\[
X_{33}(t + 4) = [x_0x_1...x_{27}]_{12:28} [0_{28:34} \mid I_{28:26}] \oplus ([x_{28}x_{29}x_{30}x_{31}] \oplus Z(t)_{1:4}) \cdot D_{4:32}
\]

\[
G = [1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]
\]

\[
D = \begin{bmatrix}
1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

The calculator schematic described by (3) is depicted in Figure 3-19, where each box represents a flip-flop and each circle represents an exclusive-or gate.
As for the CRC16 case, we construct a schematic using $k=4$ and $n=16$. The CCITT generating polynomial $g_2(x)$ is also listed in Table 3-1. Similar to the CRC32 case, we obtain the state transition equations below:

$$X_{16}(t+4)=[x_0.x_1...x_{11}]·[0_{12:4} | I_{12:12}] ⊕ ([x_{12:13}x_{14:15}] ⊕ Z(t)_{1:4}) · D_{8:16}$$

... (6)

$$G=[1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0]$$

... (7)

$$D = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

... (8)

The physical circuit following (6) is depicted in Figure 3-20. Obviously, CRC schematic using parallel architecture, instead of serial architecture, needs no S/P and P/S. It also speeds up the CRC computation without gate count expanding sharply.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Degree</th>
<th>Generating Polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1(x)$</td>
<td>32</td>
<td>$y^{32}+y^{26}+y^{23}+y^{22}+y^{19}+y^{18}+y^{16}+y^{14}+y^{10}+y^8+y^7+y^5+y^4+y^2+y^1+1$</td>
</tr>
<tr>
<td>$g_2(x)$</td>
<td>16</td>
<td>$y^{16}+y^{12}+y^8+1$</td>
</tr>
</tbody>
</table>

Table 3-1 Generating polynomial table
Chapter 4
Direct Memory Access Functions

The kernel functions of this MAC controller (regarding to IEEE 802.11) are represented in Chapter 3. In order to provide a complete and cost-effective solution for the MAC controller, a DMA controller and an Arbitrator, must also be carried out. This chapter explains the functionality of the DMA controller and ABR. The block diagram is shown as Figure 4-1. The Data Manager controls when and how the data flows among SRAM, BIU, and FIFOs. The Address Controller specifies the starting address of the data in the SRAM and also counts the length of the data.

Figure 4-1   Block diagram of DMA and ABR modules
4.1 SRAM Arrangement

As shown in Figure 4-2, we divide a 32K x8 SRAM into two regions: TX region and RX region. The TX and RX regions respectively consist of 2 units and 10 units. Each unit is 2.5K bytes in size, because the maximum MPDU length is 2346 bytes. The reason of unbalanced structure for TX region and RX region is because during transmitting, the pending data could be buffered temporarily in the host memory, while during receiving, the received data should be stored at the time of reception. We therefore devote a 10-unit buffer for frame receptions to avoid the frame loss due to buffer overrun.

The TX region (as well as the RX region) is further partitioned into two portions: one is dedicated to Management frames, and the other is used to record the Data frames. Based on the appearing ratio of frame type, the Data frame portion of RX region consumes 9 out of 10 total units.

![SRAM Partitions](image)

**Figure 4-2** SRAM (32K x8) Allocation
4.2 Memory Management

Since the priority of the Management frame is higher than the Data frame, DMA controller will process the Management frame first in both the TX and RX regions. Similar to a general ring buffer structure, DMA controller provides two sets of pointers. The first set is controlled by host software, which reflects the current access status at Host side. The second set is handled by MAC controller, which shows the access status to the SRAM at the LAN card side. In the Data_rx portion, the DASP (pointer controlled by S/W) always chases behind DAHP (pointer controlled by H/W). These pointers are illustrated in Figure 4-3.
4.3 Flowchart of the Data Manager

As shown in Figure 4-4, the Data Manager is also implemented through a Finite State Machine approach. The Data Manager negotiates among four operations: Read, Write, Transmission, and Reception. Each operation corresponds to one state. At the end of the RX State, the Append-length State will place the length information of the received frame in the first two bytes of each unit belonging to RX region.

![Flowchart of the Data Manager](image)

Figure 4-4  State diagram of DMA controller

Whenever the FIFO controller is accessing the SRAM, the system bus interface activity will be given a ready wait. Note that the RX operation, TX operation, Write operation, and Read operation can be operated in an interleaved fashion.
Chapter 5
Verification and Simulation Results

5.1 Design Flow and Tools

In order to identify the feasibility of our solution, we use the ASIC approach to carry out a FPGA sample. The entire design flow, as well as the design platforms and tools employed, are drawn in Figure 5-1. Their design details are discussed in the previous chapters. As shown in Figure 5-1, after simulating the behavioral level code, we transform the behavioral code into gate level code by means of Synopsys tool. It can be synthesized into two kinds of netlist files using respective libraries. The netlist file using Comapss 0.6um cell library is prepared for another kind of design flow (cell-based design). The other netlist using Altera library is the entry for MAX+PLUS II tool. Similar to the behavioral level code, the netlist files are verified via the gate level simulation again. After the simulation results of netlist files are identical to the behavior level simulation, we analyze the netlist timing, followed by placing & routing the layout, and programming the selected FPGA device both under the MAX+PLUS II design environment. Finally, substantiation of our design through an emulation board ends the verification process. The simulation results are classified into four levels: Behavioral level, Gate level, FPGA level, and Board level. The simulation results of different levels are presented in the subsequent sections.
Figure 5-1  Design Flow
5.2 Gate Level Simulation Results

**Reception FSM Simulation I**

Figure 5-2 illustrates how the RX FSM processes the received Probe Request frame. The RX FSM is initiated by FSM_en signal, and is ended by Length_counted message. The processing results are put into the RPI register file including Error_status, Tx_demand, and Control registers.

![RX FSM Simulation: Receiving Probe](image-url)
Reception FSM Simulation II

A more complex situation for RX FSM is shown in Figure 5-3. There are two types of frames arriving at the receiver. The first one is a Data type frame. The second one is an unrecognizable frame, since its type field value is unspecified in the standard. For an unknown frame, the Abort State enables the abort process, and issues an abort_crc32 signals to reset the related functional blocks.

Figure 5-3 RX FSM Simulation: Receiving Data and Unrecognizable
RX CRC32 Calculation Sub-module

Figure 5-4 simulates the capability of CRC32 error detection. The first frame received contains an invalid CRC32, so CRC_32_error is outputted to inform RX_FSM such status, and RX CRC32 calculator is reset by abort_crc32 input signal. The second frame has a valid CRC32, so the unique remainder pattern, i.e. c704dd7b, is obtained.

Figure 5-4  RX CRC32 Sub-module: Abort + CRC32 Error
**PLCP header Generation Sub-module**

Figure 5-5 simulates two cases: Generating PLCP header of RTS_2M frame, and Generating PLCP header of ACK_1M frame. The PLCP header generation of RTS_2M frame is initiated by En_Tx signal from TX FSM. The En_Rx signal enables the PLCP header generation of ACK_1M frame.

![Figure 5-5 PLCP header Generator: RTS_2M + ACK_1M](image-url)
MAC header Generation Sub-module

Figure 5-6 illustrates the MAC header generating procedure of CTS frame and RTS frame. The MAC header field values are displayed at the bottom of the figure.

Figure 5-6    MAC header Generator: CTS_1M + RTS_1M
Control FSM Sub-module

As shown in Figure 5-7, the system realizes that various fields are ready. Hence, as Control FSM properly transits the states, these fields are loaded sequentially onto the TxD_ctrl data bus.

Figure 5-7  Control FSM Simulation
Control Frame Generation Module

Figure 5-8 shows that Control Frame Generator generates a RTS frame. Figure 5-8 contains all signals appearing previously in Figures 5-5, 5-6, and 5-7.

Figure 5-8   Control Frame Generator: Generating RTS_2M
**DMA Control Module**

Figure 5-9 displays the simulation waveform of writing operation. The WE_host goes active low to write the data into SRAM.

![DMA Controller: Writing operation](image-url)
**ABR Module I**

Generally speaking, the ABR module is responsible for granting the bus to the system bus interface. If transmission or reception operation is initiated, the ABR switches the bus to the MAC controller upon reception of BUS_switch signal from DMA controller. In Figure 5-10, we simulate the case that the reading operation is interrupted by transmission operation.

![Simulation of ABR Module](image)

**Figure 5-10** ABR module: Reading process interrupted by Transmission
**ABR Module II**

In contrast to the previous case, Figure 5-11 represents that the writing operation is interrupted due to the occurrence of reception operation. The signals named _dma represent the inputs from DMA controller. The input signals named _host come from Host system. The output signals lying on the bottom of Figure 5-11 are the SRAM memory interface.

![Figure 5-11 ABR Module: Writing process interrupted by Reception](image-url)
5.3 FPGA Level Simulation Result

After finishing the gate-level simulation at Synopsys, we use MAX+PLUS II timing simulator to verify the FPGA sample. Figure 5-12 shows that Reception FIFO receives an ACK_1M frame.

![Figure 5-12](image_url)  
Figure 5-12  Reception FIFO simulation at FPGA-level
5.4 Area and Timing

We can estimate the area and critical timing of our netlist files by *Synopsys* for our cell-based design, and also by *MAX+PLUS II* for its equivalent FPGA counterpart. Table 5-1 lists the areas and critical clock cycle time for each module. The cell-based design uses *Compass 0.6um* cell library, and yet the FPGA design adopts *Flex 10K* library.

<table>
<thead>
<tr>
<th>Software</th>
<th>Module</th>
<th>Area</th>
<th>System Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synopsys</td>
<td>DMA Controller</td>
<td>about 3000 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ABR</td>
<td>about 450 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SEL/TX CRC32</td>
<td>about 800 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control Frame Generator</td>
<td>about 5000 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CSMA/CA</td>
<td>about 14000 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BIU</td>
<td>about 3500 gates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Top Module</td>
<td>about 27000 gates</td>
<td>20ns</td>
</tr>
<tr>
<td>MAX+PLUS II</td>
<td>Top Module</td>
<td>4600 logic cell</td>
<td>24ns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1500 FFs’</td>
<td></td>
</tr>
</tbody>
</table>

Table 5-1 Attributes of the MAC controller

The destined system clock is 44 MHz. The endurable clock rate of cell-based aspect (50 MHz) is actually higher than 44 MHz. Note that if we use 0.35um technology library instead of 0.6um technology, the system clock rate will uprise further. For the FPGA design flow, the system clock only reaches about 41 MHz, which is slower than our demand. Yet 41 MHz is high enough for our MAC controller to operate normally.
Chapter 6  Summary

6.1 Conclusions

In this thesis, we propose a new hardware architecture of MAC sublayer controller and a complete system solution for IEEE 802.11 Wireless LAN standard. The characteristics of our proposal are to build one CSMA/CA module and one Control Frame Generation module to respectively perform Multiple Access Control algorithm and control frame generation. We also verify the feasibility of our proposal by ASIC design flow. The verification and emulation results do support our goal.

We compare the performance of our solution with the available products in market. The comparison is listed in Table 6-1. The table does merit the all-hardware implementation.

<table>
<thead>
<tr>
<th>Item</th>
<th>CPU-based Architecture</th>
<th>Our proposed Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Area</td>
<td>Large</td>
<td>Small</td>
</tr>
<tr>
<td>Speed</td>
<td>Slow</td>
<td>Fast</td>
</tr>
<tr>
<td>Software programming</td>
<td>Heavy</td>
<td>Almost Effortless</td>
</tr>
</tbody>
</table>

Table 6-1  Comparison between CPU-based and our proposed architecture
6.2 Future Work

In this thesis, some IEEE 802.11 MAC sublayer functions, such as Infrastructure Access, are not taken into considerations. Some functional blocks proposed in this thesis are not implemented in the current stage of implementation. These future works are summarized in Table 6-2. Note that some tasks in the table can be easily fulfilled by revising the existing modules. For instance, one can easily enhance the CSMA/CA module to render the processes for Beacon frame and PS-Poll frame. An example is the enhancement of the Control Frame Generator for the generation of PS-Poll frame. After finishing the above two enhancements, the MAC controller can support the infrastructure access capability.

Some of the future work are really challenging, and requires careful design. The Management Frame Generation module is the most interesting one. This module independently generates partial or all subtypes of Management frames. If Management Frame Generator is implemented, the host driver only needs to process the data frames since both of the Control and Management frames are now generated and processed by hardware circuits; and hence, the effort in maintaining the driver is significantly reduced.

<table>
<thead>
<tr>
<th>Level</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Straightforward</td>
<td>Roaming Mechanism</td>
</tr>
<tr>
<td></td>
<td>Infrastructure Access</td>
</tr>
<tr>
<td></td>
<td>Higher transmission data rate (5.5M bps, 11M bps) support</td>
</tr>
<tr>
<td>Challenging</td>
<td>Management Frame Processing Module</td>
</tr>
<tr>
<td></td>
<td>Power Management Wizard</td>
</tr>
<tr>
<td></td>
<td>Double Buffering Structure</td>
</tr>
</tbody>
</table>

Table 6-2 Table of Future Work
Reference


Appendix A

Medium Access Control Chip
& Bus Interface Unit Module
Spec
A.1. Introduction

A.1.1 Synopsis of this spec

This Specification specifies the functionality of the MAC chip module, DMA module, and Bus Interface Unit (BIU) module. The data used by this chip is presumed to be stored in an external 32K SRAM.

A.1.2 Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Rev #</th>
<th>Who</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>98/08/17</td>
<td>1.0</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>802.11 TX Finite State Machine</td>
</tr>
<tr>
<td>98/09/18</td>
<td>1.1</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>FIFO design in CSMA/CA module</td>
</tr>
<tr>
<td>98/11/10</td>
<td>1.2</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>Syntax error type updates</td>
</tr>
<tr>
<td>98/12/31</td>
<td>1.3</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>Modify the Spec’s content</td>
</tr>
<tr>
<td>99/04/06</td>
<td>1.4</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>Modify the Spec’s content</td>
</tr>
<tr>
<td>99/04/22</td>
<td>1.5</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>Modify the Spec’s content</td>
</tr>
<tr>
<td>99/05/24</td>
<td>1.6</td>
<td>M.-T. Hong &amp; F.-S. LIN</td>
<td>Modify the Spec’s content</td>
</tr>
</tbody>
</table>
A.2. System Level Description

A.2.1 Feature Set

(1) BIU module acts as a data access interface between Host system and LAN card.
(2) MAC module, consisting of CSMA/CA submodule, and Control submodule and Power Control submodule, is the implementation of the IEEE 802.11, such as the random backoff mechanism and generation of IEEE 802.11 Control Frames.
(3) The data exchange scheme between Host system and LAN card is using the Direct Memory Access (DMA) which consists of one part: local DMA.

A.2.1.1 features

• Supports 1 Mbps or 2 Mbps half duplex data transmission

• Supports IEEE 802.11 CSMA/CA function (see Content 5th)

• Supports IEEE 802.11 Power Management

• Supports Ad Hoc Network (IBSS)

• PCMCIA interface (see Content 5th)

• Fast response for Control frames (see Content 5th)

• Uses a 32K x 8 SRAM for data storage (see Content 5th)
A.2.2 System Level Description

The MAC module is placed after the BIU (PCMCIA) module and before the Baseband module as shown in Figure 1.

A.2.3 Software Issues

When the Host system wants to transmit Data/Management frames, it shall enable the BIU module (certainly, after the LAN card is initialized), and transmit the Data/Management frames to the 32KB SRAM on the LAN card through the BIU module. The MAC module will then access the Data/Management frames, and transmit them to Baseband module.

According to our design, the Host system does not have to transmit the Control frames before transmitting the Data/Management frames, which is required by the IEEE 802.11 Standardization (such as RTS frame). This is because in our design, the MAC module itself will add the necessary Control frames before transmitting the Data/Management frames. In the other hand, the Host system needs to prepare the CRC16 field in PLCP Field of the transmission frames and the LAN card prepares the CRC32 field in the MAC format.

If some errors occur during the transmission of the MAC module, the MAC module will put the necessary status code into the Status/Command registers, and set the Interrupt register (Active high). The Host system, after informed by Interrupt from MAC module, should read the Status registers and respond with necessary error recovery actions (The error recovery actions of the Host system is beyond the scope of the MAC chip design).

The Host system shall assign reasonable ATIM Window, Data Length, Source Address and RTSThreshold to the corresponding registers of the MAC module, which will be referred by the operations of the MAC module.
As suggested by IEEE 802.11 Specification, the Management frame should have higher priority than the Data frames. Accordingly, if there are Management frame and Data frame in the SRAM at the same time, the MAC module will transmit the Management frame.
A.3. Interfacing Notes

A.3.1 CSMA/CA and control interface

The internal connections between the CSMA/CA submodule and the Control submodule are as follows.

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Bus Width</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>En_Tx</td>
<td>O</td>
<td>1</td>
<td>Enable Control submodule to read the TX frame status registers</td>
</tr>
<tr>
<td>En_Rx</td>
<td>O</td>
<td>1</td>
<td>Enable control submodule to read the RX frame status registers</td>
</tr>
<tr>
<td>RTS_confirm</td>
<td>I</td>
<td>1</td>
<td>Control submodule complete RTS transmission</td>
</tr>
<tr>
<td>CTS_confirm</td>
<td>I</td>
<td>1</td>
<td>Control submodule complete CTS transmission</td>
</tr>
<tr>
<td>ACK_confirm</td>
<td>I</td>
<td>1</td>
<td>Control submodule complete ACK transmission</td>
</tr>
<tr>
<td>Ctrl_retry</td>
<td>O</td>
<td>1</td>
<td>Set to 1, if transmission failed.</td>
</tr>
<tr>
<td>Add_1_Tx</td>
<td>O</td>
<td>48</td>
<td>Receiver address</td>
</tr>
<tr>
<td>Add_2_Tx</td>
<td>O</td>
<td>48</td>
<td>Source address</td>
</tr>
<tr>
<td>Power_manage</td>
<td>O</td>
<td>1</td>
<td>Power management bit: one = PS state, and zero = Awaken state</td>
</tr>
<tr>
<td>Data_rate_Tx</td>
<td>O</td>
<td>1</td>
<td>0 for 1 Mbps, 1 for 2 Mbps</td>
</tr>
<tr>
<td>Ctrl_subtype_Tx</td>
<td>O</td>
<td>4</td>
<td>Data frame subfield in MAC header. RTS, CTS, ACK</td>
</tr>
</tbody>
</table>

A.3.2 CSMA/CA and Baseband interface

The interface between the CSMA/CA submodule and the Baseband module are as follows.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Bus width</th>
<th>I/O type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tx_Data</td>
<td>4</td>
<td>O</td>
<td>Data pins. 4 bits, if using MII.</td>
</tr>
<tr>
<td>Tx_clk</td>
<td>1</td>
<td>1</td>
<td>250KHz or 500KHz</td>
</tr>
<tr>
<td>Rx_Data</td>
<td>4</td>
<td>1</td>
<td>Data pins. 4 bits, if using MII.</td>
</tr>
<tr>
<td>Rx_clk</td>
<td>1</td>
<td>1</td>
<td>250KHz or 500KHz</td>
</tr>
<tr>
<td>Length</td>
<td>16</td>
<td>1</td>
<td>44 Mhz, indicate the current MPDU length</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
<td>I/O</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------</td>
<td>-----</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Service</td>
<td>8</td>
<td>I</td>
<td>44 Mhz, reserved</td>
</tr>
<tr>
<td>CRC_16_error</td>
<td>1</td>
<td>I</td>
<td>44 Mhz, if the reception MPDU CRC16 is error</td>
</tr>
<tr>
<td>Register_en</td>
<td>1</td>
<td>I</td>
<td>44 Mhz, set to one if MAC will write the registers in Baseband</td>
</tr>
<tr>
<td>Tx_ready</td>
<td>1</td>
<td>O</td>
<td>44 Mhz, Notify Baseband module to receive data</td>
</tr>
<tr>
<td>Rx_ready</td>
<td>1</td>
<td>I</td>
<td>44 Mhz, Notify MAC module to receive data</td>
</tr>
<tr>
<td>CCA</td>
<td>1</td>
<td>I</td>
<td>44 Mhz, Set to 1, if channel is busy; otherwise, set to 0</td>
</tr>
<tr>
<td>Bb_reset</td>
<td>1</td>
<td>O</td>
<td>44 Mhz, Reset Baseband module, if necessary</td>
</tr>
<tr>
<td>Rx_sleep</td>
<td>1</td>
<td>O</td>
<td>44 Mhz, Turn off the power of the Baseband module</td>
</tr>
<tr>
<td>Sys_clk</td>
<td>1</td>
<td>I</td>
<td>System clk, 44 Mhz from baseband module</td>
</tr>
</tbody>
</table>

### A.3.3 BIU Module and Host system interface

This shows the PCMCIA Interface Pins.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Bus width</th>
<th>I/O type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>IORD_</td>
<td>1</td>
<td>I</td>
<td>Read enable for I/O function, active low</td>
</tr>
<tr>
<td>IOWR_</td>
<td>1</td>
<td>I</td>
<td>Write enable for I/O function, active low</td>
</tr>
<tr>
<td>CE1_</td>
<td>1</td>
<td>I</td>
<td>Chip enable for 8 bits data bus, active low</td>
</tr>
<tr>
<td>CE2_</td>
<td>1</td>
<td>I</td>
<td>Chip enable for 16 bits data bus, active low</td>
</tr>
<tr>
<td>OE_</td>
<td>1</td>
<td>I</td>
<td>Output enable for attribute memory, active low</td>
</tr>
<tr>
<td>WE_</td>
<td>1</td>
<td>I</td>
<td>Write enable for attribute memory, active low</td>
</tr>
<tr>
<td>REG_</td>
<td>1</td>
<td>I</td>
<td>Enable for attribute memory, active low</td>
</tr>
<tr>
<td>Reset</td>
<td>1</td>
<td>I</td>
<td>Reset pin, active high</td>
</tr>
<tr>
<td>A</td>
<td>26</td>
<td>I</td>
<td>Address bus for data</td>
</tr>
</tbody>
</table>
### A.3.4 SRAM Interface

Shown in the section 5.4.1.

### A.3.5 BIU module and CIS ROM interface

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Bus width</th>
<th>I/O type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address_CIS</td>
<td>15</td>
<td>I</td>
<td>Address bus</td>
</tr>
<tr>
<td>D_CIS</td>
<td>8</td>
<td>I/O</td>
<td>Data bus</td>
</tr>
<tr>
<td>CE_CIS</td>
<td>1</td>
<td>I</td>
<td>Chip enable for CIS, active low</td>
</tr>
<tr>
<td>OE_CIS</td>
<td>1</td>
<td>I</td>
<td>Read enable for CIS, active low</td>
</tr>
<tr>
<td>WE_CIS</td>
<td>1</td>
<td>I</td>
<td>Write enable for CIS, active low</td>
</tr>
</tbody>
</table>
A.4. Theory of operation
A.4.1 MAC MODULE Block Diagram in IBSS Network

![Diagram](image)

Figure A.1  MAC Chip Architecture

Based on the above diagram, we will describe the block functionality in details below.

I. BIU module acts as an interface between Host system and LAN card. It not only contains the card initialization information but also enables the MAC controller.

II. DMA controller handles when and how the frames transmit from SRAM to the FIFO. It uses a handshaking operation with FIFO control when transmitting the frames. DMA controller also justifies whether or not there is any frame in SRAM waiting for transmission. If so, the DMA controller will inform the CSMA/CA submodule by sending a control signal to it.

78
III. Local memory is assumed to be a 32K X 8 SRAM module. It is partitioned into two parts: Transmitting part and Receiving part, which respectively contains 2 and 10 sections, where each section is 2.5 K bytes in size. Note that the maximal size of the IEEE 802.11 Frames is 2346 bytes, which is less than 2.5K bytes.

IV. CSMA/CA submodule performs the IEEE 802.11 CSMA/CA mechanism. It contains three timers: the IFS timer, random backoff timer and ATIM timer. Its functional block diagram is shown in Figure 2.

![Figure A.2 CSMA/CA Submodule Architecture](image-url)
### A.4.2 CSMA/CA Tx MODULE Fixed Registers

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal</td>
<td>8</td>
<td>R/W</td>
<td>CTS/ACK timer</td>
</tr>
<tr>
<td>Unicast or M/B</td>
<td>1</td>
<td>R/W</td>
<td>Unicast or Broadcast/multicast</td>
</tr>
<tr>
<td>Ctrl_subtype_Tx</td>
<td>6</td>
<td>R/W</td>
<td>100000: Data frame&lt;br&gt;001000: Beacon frame&lt;br&gt;001001: ATIM frame</td>
</tr>
<tr>
<td>Retry</td>
<td>1</td>
<td>R/W</td>
<td>0: First frame, 1: Retry frame</td>
</tr>
<tr>
<td>Power_manage</td>
<td>1</td>
<td>R/W</td>
<td>0: Active mode, 1: Power save mode</td>
</tr>
<tr>
<td>Add_1_Tx</td>
<td>48</td>
<td>R/W</td>
<td>Provide for control module</td>
</tr>
<tr>
<td>Add_2_Tx</td>
<td>48</td>
<td>R/W</td>
<td>Provide for control module</td>
</tr>
<tr>
<td>Length</td>
<td>16</td>
<td>R/W</td>
<td>RTSThreshold, used to decide to send RTS or not</td>
</tr>
<tr>
<td>Fragment</td>
<td>4</td>
<td>R/W</td>
<td>zero: first MPDU in an MSDU, not zero: non-first MPDU in an MSDU</td>
</tr>
</tbody>
</table>

### A.4.3 CSMA/CA MODULE Status/Command Registers

#### A.4.3.1 Interrupt Tx Status registers

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Txend</td>
<td>2</td>
<td>R/W</td>
<td>Indicate that the Beacon, ATIM frames was successfully transmitted to Baseband module. 01: Beacon frame, 10: ATIM frame</td>
</tr>
<tr>
<td>ATIMend</td>
<td>1</td>
<td>R/W</td>
<td>Notify the Host system to stop transmitting the ATIM frames.</td>
</tr>
<tr>
<td>Data_ACK_to</td>
<td>1</td>
<td>R/W</td>
<td>If the data ACKtimer is timeout, set to one. Note that ACKtimer limit depends on the current data rate (1 or 2 Mbps).</td>
</tr>
<tr>
<td>Magn_ACK_to</td>
<td>1</td>
<td>R/W</td>
<td>If the magn ACKtimer is timeout, set to one. Note that ACKtimer limit depends on the current data rate (1 or 2 Mbps).</td>
</tr>
</tbody>
</table>
A.4.3.2 Interrupt Rx Status registers

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rxdata</td>
<td>1 bit</td>
<td>R/W</td>
<td>Indicate that a Data frame was received</td>
</tr>
<tr>
<td>Rxmag</td>
<td>1 bit</td>
<td>R/W</td>
<td>Indicate that a Management frame was received</td>
</tr>
<tr>
<td>ACKrx</td>
<td>1bit</td>
<td>R/W</td>
<td>Indicate that a ACK frame was received</td>
</tr>
<tr>
<td>Beaconrx</td>
<td>1 bit</td>
<td>R/W</td>
<td>Indicate that a beacon frame was received</td>
</tr>
</tbody>
</table>

A.4.3.3 Tx Configuration registers

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATIMtimer</td>
<td>3 bits</td>
<td>R/W</td>
<td>It is used only in IBSS network under power save mode.</td>
</tr>
<tr>
<td>PS</td>
<td>1 bit</td>
<td>R/W</td>
<td>Power save mode, 0: active mode, 1: power save mode.</td>
</tr>
<tr>
<td>RTSThreshold</td>
<td>13 bits</td>
<td>R/W</td>
<td>If the length of the Data frame is larger than RTSThreshold, the MAC module will launch the RTS/CTS handshaking; otherwise, it won’t.</td>
</tr>
<tr>
<td>Data_length_w</td>
<td>12 bits</td>
<td>R/W</td>
<td>The register stores the total length of the current transmitted Data frame</td>
</tr>
<tr>
<td>Magn_length_w</td>
<td>8 bits</td>
<td>R/W</td>
<td>The register stores the total length of the current transmitted management frame</td>
</tr>
<tr>
<td>Data_length_r</td>
<td>12 bits</td>
<td>R</td>
<td>The register stores the total length of the current received Data frame</td>
</tr>
<tr>
<td>Magn_length_r</td>
<td>8 bits</td>
<td>R</td>
<td>The register stores the total length of the current received management frame</td>
</tr>
<tr>
<td>SA</td>
<td>48 bits</td>
<td>R/W</td>
<td>Source Address</td>
</tr>
<tr>
<td>DWSW</td>
<td>1 bit</td>
<td>R/W</td>
<td>Data wrote Software Pointer</td>
</tr>
<tr>
<td>MWSP</td>
<td>1 bit</td>
<td>R/W</td>
<td>Management wrote Software Pointer</td>
</tr>
<tr>
<td>DASP</td>
<td>4 bits</td>
<td>R/W</td>
<td>Data Accepted Software Pointer</td>
</tr>
<tr>
<td>DAHP</td>
<td>4 bits</td>
<td>R/W</td>
<td>Data Accepted Hardware Pointer</td>
</tr>
<tr>
<td>MASP</td>
<td>1 bit</td>
<td>R/W</td>
<td>Management Accepted Software Pointer</td>
</tr>
</tbody>
</table>
A.4.4 CSMA/CA Rx MODULE Fixed Registers

A.4.4.1 TX Demand Register between RX_FSM and Tx_FSM

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATIM_received</td>
<td>1</td>
<td>R/W</td>
<td>Inform TX_FSM about receiving ATIM</td>
</tr>
<tr>
<td>CTS_received</td>
<td>1</td>
<td>R/W</td>
<td>Inform TX_FSM about receiving CTS</td>
</tr>
<tr>
<td>ACK_received</td>
<td>1</td>
<td>R/W</td>
<td>Inform TX_FSM about receiving ACK</td>
</tr>
<tr>
<td>Beacon_received</td>
<td>1</td>
<td>R/W</td>
<td>Inform TX_FSM about receiving Beacon</td>
</tr>
</tbody>
</table>

A.4.4.2 RX_FSM’s Error Status Register

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence_number_error</td>
<td>1</td>
<td>R/W</td>
<td>Data Frame out of order</td>
</tr>
<tr>
<td>Frag_number_error</td>
<td>1</td>
<td>R/W</td>
<td>Data Frame out of order</td>
</tr>
<tr>
<td>Unknown_manage subtype</td>
<td>1</td>
<td>R/W</td>
<td>Unsupported Management subtype</td>
</tr>
<tr>
<td>Unknown_data_subtype</td>
<td>1</td>
<td>R/W</td>
<td>Unsupported Data subtype</td>
</tr>
<tr>
<td>Unknown_control subtype</td>
<td>1</td>
<td>R/W</td>
<td>Unsupported Control subtype</td>
</tr>
<tr>
<td>CRC_16_error</td>
<td>1</td>
<td>R/W</td>
<td>PHY CRC check error</td>
</tr>
<tr>
<td>CRC_32_error</td>
<td>1</td>
<td>R/W</td>
<td>MAC CRC check error</td>
</tr>
<tr>
<td>First_frag_interleaving error</td>
<td>1</td>
<td>R/W</td>
<td>Protect another STA’s first fragment to insert the original fragment exchange sequence</td>
</tr>
<tr>
<td>Non_first_interleaving error</td>
<td>1</td>
<td>R/W</td>
<td>Protect another STA’s non-first fragment to insert the original fragment exchange sequence</td>
</tr>
</tbody>
</table>
### A.4.4.3 MAC Configure Register

<table>
<thead>
<tr>
<th>Name</th>
<th>Bus</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Board_test_mode</td>
<td>1</td>
<td>R/W</td>
<td>Convenient to do on board test</td>
</tr>
<tr>
<td>Override_mode</td>
<td>1</td>
<td>R/W</td>
<td>Ignore overall packet error; For monitoring LAN purpose</td>
</tr>
<tr>
<td>BSSID</td>
<td>48</td>
<td>R/W</td>
<td>Indicating current attached BSS</td>
</tr>
<tr>
<td>SA</td>
<td>48</td>
<td>R/W</td>
<td>The LAN card number</td>
</tr>
<tr>
<td>Power_management</td>
<td>1</td>
<td>R/W</td>
<td>Current STA’s power mode</td>
</tr>
</tbody>
</table>
A.5. Unit Operation

A.5.1 CSMA/CA Submodule

The function of each unit in CSMA/CA submodule is described below.

1. 802.11 mechanism unit is the most important one in performing the IEEE 802.11 protocol (CSMA/CA). This unit is roughly divided into four state machines: Idle, CCA, TX, and RX. When this unit is in operation, it will obtain information from other units, such as Random timer unit and DIFS Timer unit. In addition, this unit uses the PS (Power Save Mode) to decide whether to enter the Power save mode or not.

2. ATIM counter is used when the station is in power save mode. As specified in IEEE 802.11, the PS station will maintain the ATIM timer during the period between Beacons. During the ATIM time period, every station will awake, and check whether or not there is a frame for itself. If there is any, the PS station will keep awaking to receive those frames. If the PS station wants to transmit data, it will first transmit the ATIM frame to contend for the transmission right during the ATIM timer period. In summary, the ATIM counter unit is to ensure that every station will awake at the right time, and the transmission frames will not be lost.

3. DIFS Timer is to count the DIFS. It is about 50 us in length. Similarly, Slot time unit is used to count the slot time (20 us).

4. Retry count unit is to calculate the numbers of the re-transmission. This number is also used in calculating the random backoff contention window, which increases exponentially with respect to retry count.

5. Random Backoff Mechanism defers the transmission for a predefined time when the system wants to transmit data. It will choose different ways to compute the deferred time according to the frame type: i.e., Data frame, Beacon frame or
ATIM frame. For Data frames, the contention window varies from 0 to CW (7-255) time slots. For Beacon transmission, the contention window varies between 0 and 2Cwmin (14) time slots. As for ATIM frames, the contention window lies within 0 and CWmax (255) time slots. Note that a time slot is 20us in length.

6. Tx FIFO and RX FIFO are 48-byte and 1-byte respectively in length. The FIFO control is to transmit data from SRAM to FIFO, and at the same time, count the number of data current in the FIFO in order to prevent from data overrun.

What follows is the flow chart performed in our MAC chip. (Cf Section 1 and Section 2, also Figure A.2)

I.1 Section 1: 802.11 protocol Tx flow chart in CSMA/CA submodule

Data frame flow chart (subfield = 100000):

![Data Frame flow chart](image)

Figure A.3 Data Frame flow chart.

For a Data frame, 802.11 mechanism unit will first check whether the fragment
number of that frame is zero or not.

If the fragment number is zero, 802.11 protocol mechanism will be performed the flow chart as shown in Figure 3.2a, which is exactly the 802.11 CSMA/CA mechanism. It includes DIFS timer and Random back-off timer. In the initial state, if some other station is currently transmitting data (under which the CCA should be one), then the station will wait for the next contention period, and the detect bit will be set to one as shown in Figure A.3. In the next contention period, it will use the random timer to reduce the collision probability in case many stations want to contend for the media at the same time. Within the random back time, if some other station get the right to transmit, and hence, starts to transmit data (under which the CCA should be one), the random timer will stop. The random timer will continue to decrease in the next contention period. If the random timer is decreased to zero (and hence, the station gets the right to transmit), it will justify whether to transmit RTS or not by RTSthreshold stored in the registers (default value is 3000 bytes). If RTS frame is successfully transmitted, 802_11 mechanism unit will inform FIFO control unit to transmit data from FIFO to Baseband module in 4-bit wide MII bus. The detail of FIFO transmission will be discussed in the Section 2.

If the fragment number is not zero, we will transmit the next fragment (frame) as soon as it could.

② Beacon frame flow chart (subfield = 001000):
Figure A.4  Beacon transmission flow chart.

This flow chart describes another frame operation. Beacon frame is transmitted in every beacon interval indicating by TSF timer (default is 100 Kus). Before transmission, the system will justify whether the station is in power save or in awaken state. If it is in power save state, it will enable ATIM timer. If, in addition, it has data to transmit, it shall transmit ATIM request to the destination station and wait for the acknowledgement until the expiration of the ATIM window (default is 4 Kus). In such case, the system will keep awaking until the next beacon period.

If it is in awaken state, it will not start the ATIM timer. Instead, it transmits its own Beacon frame at each beacon interval. In case that the Beacon frame is successfully transmitted, the system will become a temporary Access point (AP); otherwise, it will not transmit its Beacon frame. Note that the latter situation happens when the system receives Beacon frame from some other station before the expiration of its random timer.
ATIM frame flow chart (subfield = 001000):

![ATIM frame flow chart](image)

Figure A.5 ATIM transmission flow chart.

ATIM frame is a frame that is used only in power save mode under ad hoc network. If receiving ATIM from some other station within the ATIM window, the LAN card will stop the random timer. The random timer will continue to decrease in the next transmission time (previous ATIM has been acknowledged). When its random timer decreases to zero, the LAN card will transmit its ATIM. When receiving the ATIM, the LAN card must decide whether the ATIM is for itself or not. Furthermore, the LAN card decides whether it is unicast or multicast. For a unicast ATIM, the LAN card will respond with an ACK; otherwise, the LAN card will continue to decrease the ATIM random back-off timer.

1.2 Section 1: 802.11 protocol Rx flow chart in CSMA/CA submodule

When receiving the frame without the PLCP header removed by the Baseband module, the RX_FSM will remove the MAC header, and store the remaining part of the frame into the registers defined above.
Figure A.6  RX_FSM.

The Figure A.6 is the main part of the RX_FSM in the CSMA/CA module. It describes not only STATE transition but also the each statement within every STATE. The RX_FSM is enabled by Rx_request, which are FSM_en from RX FIFO H/W or Rx_req command from S/W. If any error in each statement happens, it will transit into Abort STATE to discard the received frame. If the CRC_16 is correct, it will check the version and Address 1 fields of the MAC header to determine the received frame is Broadcast or Unicast or Not_me. The detail of Broadcast STATE is Figure A.10. If it transits into Unicast STATE, according to the type field value of the received frame, it will enter Data or Ctrl or Manage STATE respectively. The details of Data, Ctrl and Magm STATES are Figure A.7, 8, 9. After the CRC_32 field of the received frame is calculated without error, it will return to Idle STATE to wait for next frame.
Figure A.7  Data STATE of RX_FSM.

Figure A.8  Control STATE of RX_FSM.
If the received frame is the control frame, it will transit into Ctrl STATE. Because the control frame contains only few MAC header, the CRC_32 check will be done faster than Data and Management type frames. The STATE first examines the correctness of the CRC_32, and secondly it transfers the 8-bit SIGNAL register given by Baseband chip into 1-bit Data_rate_Rx register for Ctrl generation module to determine the SIGNAL field of response control frame. The third, it compares the received subtype field value with RTS subtype value, because in the Ad-hoc mode only RTS, CTS and ACK three kinds of control frames will be supported and only receiving RTS needs responding (i.e enable the Ctrl generation module).

![Flowchart of RX FSM](image)

**Figure A.9** Management STATE of RX_FSM.

In the Manage STATE, the most statements are similar to the statements of the Data STATE. Note that when receiving ATIM and Probe response frames, Control frame generator will be enabled to generate ACK frames. Other subtypes of management frames do not need acknowledgements.
Figure A.10 Broadcast and Multicast STATE of RX_FSM.

The BC/MC STATE needs a special conditional judgement by using Address 2 field (source address of the received frame) to filter out the owner Broadcast or Multicast frame.
II.1 Section 2: Transmission FIFO Design in CSMA/CA submodule

![Diagram of TX FIFO Design]

The Tx FIFO is 8-bit wide and 48 bytes in length. Data/MPDU from SRAM will be sent to the FIFO. After its completeness of pumping data to the FIFO from SRAM, the LAN card will send a control signal to notify the FIFO controller to start transmitting the data from FIFO to MII (4 bits). The clock cycle of such data transmission is provided by the Baseband module through Txclk. Therefore, the Txclk decides the data rate. If the data rate is 1 Mbps, the Txclk shall be 250 KHz. If the data rate is 2 Mbps, the Txclk shall be 500 KHz. The sysclk is 44 MHz. The sysclk is also provided by the Baseband module.
II.2 Section 2: Reception FIFO Design in CSMA/CA submodule

![Diagram of Rx FIFO Unit]

As shown in the Figure 12, we will know that the Rx FIFO is a 1 byte register in compared with the 48 bytes-TxFIFO. This is because we must wait the Rx_Data by the Rxclock, which is smaller than the systeclck. In the Rx FIFO design, there will have a 22 bytes registers to store some information from the Rx frame’s MAC header and 802_11 mechanism unit (Figure 2) will use these information. At the same time we will use the duration field in grabbed registers to start the NAV timer. If the NAV timer is over, it will inform NAVend signal to Tx_FSM in 802_11 mechanism unit (Figure 2). In the Rx_FIFO design, Rx_Ready is an important signal from the Baseband module. If the Rx_Ready is active, then we will receive Rx_Data. In addition, if we grab the total 22 bytes information from MAC header, we will inform a FSM_en signal to 802_11 mechanism unit (Figure 2) and the 802_11 mechanism unit will start to execute its function. Finally, if the reception errors occur, 802_11 mechanism unit will send an abort signal to reset the whole Rx_FIFO Unit.
A.5.2 DMA control module

![DMA block diagram](image)

Figure A.13 DMA block diagram

The main purpose of the DMA module is to transmit/receive data from SRAM to FIFO or from FIFO to SRAM. As shown in the Figure 13, the data transmission or reception is performed by 3 submodules.

First, the data controller controls when or how the data flows from SRAM to FIFO or from FIFO to SRAM. The TX data length will be stored in the configuration registers. The Rx data length will be added in the front of the MPDU in SRAM and the Rx length is two bytes.

Secondly, the address counter and controller will decide the start address of the data in SRAM and count the MPDU length.

The third controller is an interface between CSMA/CA submodule (protocol and FIFO controller) and DMA module. It also needs to prevent from FIFO overrun, and to control the signal handshaking.

The followings are the detailed functions for DMA module:

1. 8 bits bus from SRAM to FIFO:
In the initial state, the DMA controller will send the MPDU to FIFO within a Txclk because the sysclk is faster than Txclk. If the txstart for FIFO to MII is enabled, the FIFO control will move the data from FIFO to MII until the whole MPDU is transmitted, and then wait for the acknowledgement of that MPDU.

II. DMA counter and Pointer:

Because the Rx frames in SRAM are either data or management frames, therefore, we will have two pointers and one flag. That is, the two pointers are for Data frames, and the flag is to indicate the vitality of the management frames. Besides, each of the DMA module and Host system has a pointer to point to the current frame in SRAM. When the DMA module moves the frame from SRAM to FIFO, there also must be a counter to monitor the number of data bytes moved. If the counter is equal to the value in length field (the value will be stored in a register first) of the MPDU, we will know that this is the last byte of the MPDU. Then TX complete control signal will be sent to 802.11 mechanism unit (Figure 2). Therefore, we can move the next MPDU after we receive the previous ACK.
A.5.3 BIU control module

As shown in the figure, the BIU module includes address decoder and combinational control logic. Its interface to the Host system follows PCMCIA, which contains the initial setup (decode CIS) and I/O function (decode registers and SRAM). The address decoder consists of four parts.

The configuration registers store the initial values about 802.11 protocol, such as RTSThreshold value, ATIM window size, etc. These values are shown in the previous defined registers.

The SMI (Status, Command, and Interrupt) registers that store the important information to notify the Driver or the MAC chip, such as transmit complete, transmit error and transmit request.

The CIS that stores the initialization and setup information of the LAN card, such as card ID and the manufacturer ID, etc.
On the other hand, the combination logic assigns some control signals according to inputs from Host system. In summary, the BIU module performs the Wireless LAN card interface functionality. According to PCMCIA standardization, there are 26 address pins, but we only use 16 of them. Details will be shown below.

<table>
<thead>
<tr>
<th>Reserved</th>
<th>SRAM / CIS / Con_Reg</th>
<th>Address 15-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address 17-25</td>
<td>Address 16</td>
<td></td>
</tr>
</tbody>
</table>

Figure A.15 Address decode format

The I/O port Address uses the 280H, 281H, and the 282H. The 280H is for the Address[7:0] and the 281H is for the Address[15:0]. Finally, the 282H is for the data bus.

If the Address[15] is one, it represents the Address is for SRAM address, otherwise, it represents the CIS or other Registers.

The followings are the relative address locations for the Configuration registers and the CIS data.

<table>
<thead>
<tr>
<th>Address location</th>
<th>Registers name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1FFH</td>
<td>CIS Data</td>
</tr>
<tr>
<td>200H</td>
<td>Configuration Option Register</td>
</tr>
<tr>
<td>202H</td>
<td>Card Configuration and Status Register</td>
</tr>
<tr>
<td>204H</td>
<td>Pin Replacement Register</td>
</tr>
<tr>
<td>206H</td>
<td>Socket and Copy Register</td>
</tr>
<tr>
<td>208H</td>
<td>Status Tx Register</td>
</tr>
<tr>
<td>20aH</td>
<td>Status Rx Register</td>
</tr>
<tr>
<td>20eH</td>
<td>RTThreshold[7:0]</td>
</tr>
<tr>
<td>210H</td>
<td>RTThreshold[15:8]</td>
</tr>
<tr>
<td>212H</td>
<td>Tx_Data_Length[7:0]</td>
</tr>
<tr>
<td>214H</td>
<td>Tx_Data_Length[15:8]</td>
</tr>
<tr>
<td>216H</td>
<td>Tx_Magm_Length[7:0]</td>
</tr>
</tbody>
</table>
### Table A.1 Address Location for Registers within BIU module

Table 2 is the function for reading or writing the Attribute memory or the I/O function.

<table>
<thead>
<tr>
<th>Function mode</th>
<th>REG</th>
<th>CEI</th>
<th>IOWR</th>
<th>IORD</th>
<th>A0</th>
<th>OE</th>
<th>WE</th>
<th>D[7:0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Memory Read</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>Even Byte</td>
</tr>
<tr>
<td>Attribute Memory Write</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>Even Byte</td>
</tr>
<tr>
<td>I/O Read Even Byte</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>Even Byte</td>
</tr>
<tr>
<td>I/O Read Odd Byte</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>Odd Byte</td>
</tr>
<tr>
<td>I/O Write Even Byte</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>Even Byte</td>
</tr>
<tr>
<td>I/O Write Odd Byte</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>Odd Byte</td>
</tr>
</tbody>
</table>

Table A.2, Read and Write function for I/O and Attribute Memory
A.5.4 SRAM module

A.5.4.1 PIN DESCRIPTION

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0-A14</td>
<td>I / O</td>
<td>Address inputs</td>
</tr>
<tr>
<td>D0-D7</td>
<td>I / O</td>
<td>Data inputs/outputs</td>
</tr>
<tr>
<td>CS</td>
<td>I</td>
<td>Chip enable</td>
</tr>
<tr>
<td>WE</td>
<td>I</td>
<td>Write enable</td>
</tr>
<tr>
<td>OE</td>
<td>I</td>
<td>Output enable</td>
</tr>
<tr>
<td>VDD</td>
<td>I</td>
<td>Power supply</td>
</tr>
<tr>
<td>VSS</td>
<td>I</td>
<td>Ground</td>
</tr>
</tbody>
</table>

FUNCTION DESCRIPTION

![Figure A.16 SRAM (32 K x 8) Partition]

As shown in the Figure 16, we will divide the SRAM into two portions: Tx portion and Rx portion. The Tx and Rx portions respectively consist of 2 units and 10 units for which each unit is 2.5K bytes in length. The main reason for unbalanced partitions for transmitting and receiving is because during transmission, the system
can store the transmitted data in the Host memory, while during receiving, it is better to store the received data immediately as they are received in order to prevent from the frame loss.

The first portion will be further divided into two sub-portions: one is dedicatedly for Management frame, and the other, for Data frame. Since the priority of the Management frame is higher than the data frame according to IEEE 802.11, we will transmit the Management frame instead of the Data frame if both of them are in the SRAM at the same time. Therefore, we will have a register to store the frame’s information that is to justify whether the packet is data or management packet.

Finally, it needs to be pointed out that the access time of SRAM is assumed to lie between 10ns and 20ns, and the sysclk used by MAC chip is 44 MHz.
A.5.5 Control Frame Generation module

A.5.5.1 Interface between control module and baseband module

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bus</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX_CLK</td>
<td>1</td>
<td>I</td>
<td>Reference clock of TXD provided by BB</td>
</tr>
<tr>
<td>TXD</td>
<td>4</td>
<td>O</td>
<td>Data pins</td>
</tr>
<tr>
<td>MCLK</td>
<td>1</td>
<td>I</td>
<td>System clock provided by BaseBand</td>
</tr>
<tr>
<td>TX_START</td>
<td>1</td>
<td>O</td>
<td>Enable baseband to transfer data</td>
</tr>
</tbody>
</table>

A.5.5.2 Interface between control module and CSMA/CA module

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bus</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>En_Tx</td>
<td>1</td>
<td>I</td>
<td>Enable Control Module by TX_FSM</td>
</tr>
<tr>
<td>En_Rx</td>
<td>1</td>
<td>I</td>
<td>Enable Control Module by RX_FSM</td>
</tr>
<tr>
<td>ctrl</td>
<td>1</td>
<td>I</td>
<td>Reset</td>
</tr>
<tr>
<td>RTS_confirm</td>
<td>1</td>
<td>O</td>
<td>RTS was transmitted</td>
</tr>
<tr>
<td>Data_rate_TX</td>
<td>1</td>
<td>I</td>
<td>PLCP SIGNAL field values from TX_FSM</td>
</tr>
<tr>
<td>Data_rate_RX</td>
<td>1</td>
<td>I</td>
<td>PLCP SIGNAL field values from RX_FSM</td>
</tr>
<tr>
<td>Ctrl_subtype_TX</td>
<td>4</td>
<td>I</td>
<td>MAC_header Subtype field from TX_FSM</td>
</tr>
<tr>
<td>Ctrl_subtype_RX</td>
<td>4</td>
<td>I</td>
<td>MAC_header Subtype field from RX_FSM</td>
</tr>
<tr>
<td>Power_manage</td>
<td>1</td>
<td>I</td>
<td>Power Management field value</td>
</tr>
<tr>
<td>More_frag</td>
<td>1</td>
<td>I</td>
<td>For calculating ACK’s DURATION field</td>
</tr>
<tr>
<td>AID</td>
<td>16</td>
<td>I</td>
<td>MAC_header AID field values</td>
</tr>
<tr>
<td>Add_1_TX</td>
<td>48</td>
<td>I</td>
<td>MAC_header Add_1 field from TX_FSM</td>
</tr>
<tr>
<td>Add_2_TX</td>
<td>48</td>
<td>I</td>
<td>MAC_header Add_2 field from TX_FSM</td>
</tr>
<tr>
<td>Add_2_RX</td>
<td>48</td>
<td>I</td>
<td>MAC_header Add_2 field from RX_FSM</td>
</tr>
</tbody>
</table>
A.5.5.3 Control Frame module architecture

- Preamble generator:
  SYN + SFD fields are constant

Figure A.17  PLCP Frame Generator
A.5.5.4 MAC and PLCP Header generators

![Diagram of control frame selection]

Figure A.18  Control Frame Selection

A.5.5.5 Control frame module description

It is necessary to respond to the Control frames (such as RTS frames, Data frames, ATIM frames and Probe Response frames) within the SIFS interval which is suggested to be 10us in IEEE 802.11. In our design, the Control Frame Module will process the responding to the Control frames in order to shorten the SIFS interval, and hence, to speed up the overall MSDU transmission. This is the main purpose for the design of this Control Frame Module.

The Control Frame Module shall generate the complete control frame including Preamble, the PLCP header and MAC header. Thus, this module needs CRC_32 and CRC_16 generators.
The CRC_32 generator polynomial is

\[ G(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1. \]

The CRC_16 generation polynomial is

\[ G(X) = X^{16} + X^{12} + X^5 + 1 \]

The initial condition of the CRC_32 calculation is FFFF_FFFFH; and the final remainder of the CRC_32 operation is C704_DD7BH. The CRC circuits adopted here are not conventional serial architecture. These adopt parallel I/O architecture for compatible with MII interface (4-bit data pins).