

國立交通大學

電機資訊學院 資訊學程

碩 士 論 文

802.11 媒體控制器之頻道速率 WEP 和 DCF

的 ASIC 實現



Channel-Speed WEP and DCF ASIC Realization for IEEE 802.11 MAC

研 究 生： 謝才俊

指 導 教 授： 林盈達 教授

簡榮宏 教授

中 華 民 國 九 十 三 年 六 月

802.11 媒體控制器之頻道速率 WEP 和 DCF 的 ASIC 實現
Channel-Speed WEP and DCF ASIC Realization for IEEE 802.11 MAC

研究生：謝才俊

Student：Tsai-Chun Hsieh

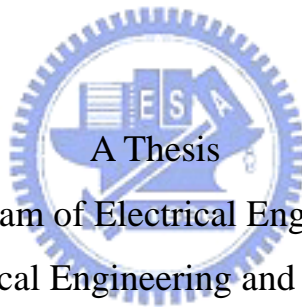
指導教授：林盈達

Advisor：Prof. Ying-Dar Lin

簡榮宏

Prof. Rong-Hong Jan

國立交通大學
電機資訊學院 資訊學程
碩士論文



Submitted to Degree Program of Electrical Engineering Computer Science
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

Computer Science

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

學生：謝才俊

指導教授：林盈達 教授

簡榮宏 教授

國立交通大學電機資訊學院 資訊學程（研究所）碩士班

摘 要

目前有內建無線網路功能的設備愈來愈普遍，而且傳輸速度也比以前都來的更快。為了實現更高吞吐量的設計，以硬體為主的設計將更常被採用。

在這篇論文當中，我們提出符合 IEEE 802.11 標準的硬體式媒體控制器及一硬體式 WEP 加解密引擎。這兩者可以非常容易的整合進入內建無線網路功能的系統晶片(SoC)當中。並且透過一先進先出的記憶體及 PLCP 表頭的前奏時間來解決 RC4 演算法中執行 金鑰排程(Key Schedule Algorithm, KSA)時所消耗掉的時間。如此將可以在 WEP 加解密功能打開下，802.11 MAC 仍然可以維持 54Mbps 頻道速率的吞吐量。有關模擬環境方面，我們設計一個虛擬 802.11 媒體控制器用來產生及檢查 802.11 的封包以驗證我們的設計。在對基頻處理器的介面訊號方面，我們選擇 Intersil HFA3861B 及 HFA3683 做為我們的實體層。

最後我們將完成支援 DCF 的媒體控制器及一具有 54Mbps 頻道速率吞吐量的 WEP 安全引擎。所有的設計使用 Verilog 語言來完成。這個設計可已完全被合成為電路。而且在使用 Xilinx FPGA (v2000efg1156-8)元件下，其操作頻率可以達到 44 MHz。消耗的邏輯閘數是 51,029 閘。

Channel-Speed WEP and DCF ASIC Relization for IEEE 802.11 MAC

Student : Tsai-Chun Hsieh

Advisors : Prof. Ying-Dar Lin

Prof. Rong-Hong Jan

Degree Program of Electrical Engineering Computer Science
National Chiao Tung University

ABSTRACT

Embedded wireless communication devices nowadays are getting more and more popular and running at higher speeds than ever before. In order to achieve the desired throughput, the hardware approach is being adopted more often.

In this thesis, we propose a hardware-based MAC IP that is compatible with IEEE 802.11 standard, and a hardware-based WEP IP. They both can be easily integrated into an SoC chip for embedded wireless communication devices. It uses FIFO RAM and PLCP preamble time to recover the overhead of processing KSA at RC4. Hence it can retain the throughput at 54Mbps Channel-speed with the WEP function turned on. In our simulation environment, a pseudo model of 802.11 MAC is designed to generate and check 802.11g frames to verify our implementation. For the baseband interface signals, we use the Intersil HFA3861B and HFA3683 as the physical layer.

Finally, we accomplish an 802.11 MAC that support DCF machine and 54Mbps Channel-speed WEP security machine. All designs are accomplished by the Verilog RTL language. It is fully synthesizable and its operation frequency can scale up to 44MHz at the Xilinx FPGA (v2000efg1156-8) device. The total equivalent gate count is 51,029.

誌 謝

感謝交通大學提供碩士在職專班學程，讓我在職場工作多年之後能夠回到學校，重拾書本吸取新的知識。同時在學習的過程當中，透過和學校老師間的互相討論，也瞭解到學術界中對相關問題的看法，對於我在工作上有許多的幫助。

感謝林盈達老師對於我學位論文的要求及指導。讓我學習到如何有系統有組織地去找問題，描述問題，進而清晰地說明如何解決這個問題。老師的叮嚀「一圖，二表，三文字」學生謹記在心。

最後感謝我的家人在我求學及做論文的過程當中，犧牲了許多寶貴的相處時間，同時也影響了你們的生活品質。因為有了你們的支持，我才能夠完成學業。

謝才俊

07/26/2004

Channel-Speed WEP and DCF ASIC Realization for IEEE 802.11 MAC

Chinese Abstract	i
English Abstract	ii
Thanks	iii
Contents	iv
List of Tables	vii
List of Figure	viii
Chapter 1 Introduction	1
Chapter 2 IEEE 802.11 MAC Protocol	4
2.1 802.11 Wireless LAN Introduction	4
2.1.1 802.11 Wireless LAN architecture	5
2.1.1.1 IBSS (Independent Basic Service Set)	5
2.1.1.2 BSS (Basic Service Set)	6
2.1.2 802.11 Wireless LAN Standard	6
2.2 Distributed Coordinated Function (DCF)	8
2.2.1 MAC Framing	8
2.2.2 CSMA/CA + ACK Mechanism	10
2.2.3 RTS/CTS Mechanism	11
2.2.4 Synchronization	12
2.2.5 Random Back-Off Mechanism	12
2.3 802.11 Security Mechanism	14
2.3.1 WEP Algorithm	14

Chapter 3 Implementation	17
3.1 Design issues and solutions	18
3.2 System Architecture Design	20
3.2.1 Chip Specification	21
3.2.2 Chip Pin Assignment	22
3.2.3 Sub-Module Partition	22
3.3 DCF Implementation	24
3.3.1 Receive Module	25
3.3.1.1 chstate Module	26
3.3.1.2 valmpdu Module	28
3.3.1.3 rx_co Module	29
3.3.1.4 chkpkt Module	30
3.3.2 Transmit Module	31
3.3.2.1 tx_co Module	32
3.3.2.2 ackpkt Module	33
3.3.2.3 rtspkt Module	33
3.3.2.4 ctspkt Module	34
3.3.2.5 back off Module	35
3.3.3 Time Stamp Module	35
3.3.4 Beacon control Module	36
3.4 WEP Algorithm Implementation	38
3.5 BBP Interface Implementation	41
3.6 RF Interface Implementation	48
Chapter 4 Function Simulation and Verification	53
4.1 802.11 MAC Simulation Environment and Test Plan	53

4.2 DCF Receive Function Verification	57
4.2.1 Receive a DATA (DA==Network Address) packet.....	57
4.2.2 Receive a RTS (DA==Network Address) packet.....	58
4.2.3 NAV Timer.....	59
4.2.4 Receive a Beacon packet in BSS mode	59
4.3 DCF Transmission Function Verification	60
4.3.1 Transmit a DATA packet.....	60
4.3.2 Transmit a Data Packet using RTS Mechanism.....	61
4.3.3 Abort the transmitted packet when lose ACK packet	62
4.3.4 Re-transmit a packet when channel is BUSY	62
4.3.5 Transmit a broadcast packet	63
4.3.6 Transmit a Beacon packet.....	64
4.4 BBP Interface Function Verification.....	64
4.4.1 Write BBP Register.....	64
4.4.2 Read BBP Register (RSSI Reg).....	65
4.5 RF Interface Function Verification.....	66
4.6 WEP Function Verification	66
4.6.1 RC4 Algorithm Verification	66
4.6.2 WEP Verification.....	68
Chapter 5 FPGA Synthesis and P&R	70
5.1 FPGA Synthesis and P&R	70
5.2 FPGA Synthesis and Timing Report.....	71
Chapter 6 Conclusion	73
Reference	74
Appendix A Boundary signals of chip.....	76

List of Tables

Table 2-1	802.11 Working Group	7
Table 2-2	Back-off Algorithm	13
Table 2-3	WEP KSA Pseudo code	15
Table 2-4	WEP PRGA Pseudo code	16
Table 3-1	Example of LENGTH calculations for CCK mode	45
Table 3-2	3 Example of LENGTH calculations under PBCC mode	46
Table 3-3	Register definition of HFA 3683A	50
Table 4-1	Test Plan	57
Table 4-1	WEP Round 1	67
Table 4-2	WEP Round 2	67
Table 5-1	v2000efg1156-8 Synthesis Report	72
Table 5-2	v2000efg1156-8 Timing Report	72



List of Figures

Figure 2-1 IBSS Mode-----	6
Figure 2-2 BSS Mode-----	6
Figure 2-3 802.11 Frame Format -----	9
Figure 2-4 RTS Frame Format-----	9
Figure 2-5 CTS Frame Format-----	9
Figure 2-6 ACK Frame Format -----	9
Figure 2-7 Beacon Frame Format -----	9
Figure 2- 8 NAV Timing-----	12
Figure 2- 99 DSSS contention window size -----	13
Figure 3-1 Design Flow Chart -----	18
Figure 3- 2 System architecture block Diagram-----	21
Figure 3- 3 Function block Diagram-----	23
Figure 3- 4 Sub module hierarchical diagram-----	24
Figure 3- 5 receive Module Interface-----	25
Figure 3- 6 Receive Function Block Diagram-----	26
Figure 3- 7 chstate Module Interface -----	27
Figure 3- 8 chstate Module state machine flow chart-----	27
Figure 3- 9 valmpdu Module Interface -----	28
Figure 3- 10 valmpdu Module state machine flow chart-----	29
Figure 3- 11 rx_co Module Interface-----	29
Figure 3- 12 rx_co Module state machine flow chart-----	30
Figure 3- 13 chkpkt Module Interface -----	31
Figure 3- 14 Transmit Function Block Diagram-----	32
Figure 3- 15 tx_co Module Interface-----	32
Figure 3- 16 ACKPkt Module Interface -----	33

Figure 3- 17 RTSPkt Module Interface -----	34
Figure 3- 18 CTSPkt Module Interface -----	34
Figure 3- 19 back off Module Interface -----	35
Figure 3- 20 tsf Module Interface -----	36
Figure 3- 21 bcncntrl Module Interface -----	37
Figure 3-22 WEP Encryption Timing -----	38
Figure 3-23 WEP Decryption Flow -----	39
Figure 3- 24 WEP Module Interface -----	40
Figure 3- 25 WEP Module state machine flow chart -----	41
Figure 3- 26 HFA3861B Pin Assignment Diagram -----	41
Figure 3- 27 HFA3861B control port read timing -----	42
Figure 3- 28 HFA3861B control port write timing -----	43
Figure 3- 29 HFA3861B TX port timing -----	43
Figure 3- 30 HFA3861B RX port timing -----	43
Figure 3- 31 mitop Module Interface -----	44
Figure 3- 32 rwbbp Module state machine flow chart -----	47
Figure 3-33 PLCP Header -----	47
Figure 3- 34 micntrl Module state machine flow chart -----	48
Figure 3- 35 HFA3683A Pin Assignment Diagram -----	49
Figure 3- 36 HFA3683A serial data input timing -----	50
Figure 3- 37 rfif Module Pin Assignment -----	51
Figure 3- 38 rfif Module FSM flow chart (0) -----	52
Figure 3- 39 rfif Module FSM flow chart (1) -----	52
Figure 4- 1 Testing Environment -----	54
Figure 4- 2 Pin Assignment of BBPMAC.v -----	54
Figure 4- 3 Receive function of BBPMAC.v -----	55

Figure 4- 4 Transmit function of BBPMAC.v -----	55
Figure 4- 5 Received a DATA packet-----	58
Figure 4- 6 Receive a RTS packet (DA == Network Address)-----	58
Figure 4- 7 Receive a RTS packet (DA != Network Address) -----	59
Figure 4- 8 Received a Beacon packet in BSS station mode -----	60
Figure 4- 9 Transmit a DATA Packet-----	60
Figure 4- 10 Transmit a Data Packet using RTS Mechanism -----	61
Figure 4- 11 Abort the transmitted packet when lose ACK packet -----	62
Figure 4- 12 Re-transmit a packet when channel is BUSY-----	63
Figure 4- 13 Transmit a Broadcast Packet -----	63
Figure 4- 14 Transmit a Beacon Packet -----	64
Figure 4- 15 BBP register writes timing -----	65
Figure 4- 16 BBP register read timing -----	65
Figure 4- 17 RF chip control timing -----	66
Figure 4- 18 RC4 KSA -----	68
Figure 4- 19 Waveform of WEP operation-----	69
Figure 5-1 FPGA Place & Route Flow Chart -----	70

Chapter 1

Introduction

Embedded wireless communication devices nowadays are getting more popular and running at higher speeds than ever before. In order to achieve the throughput of design, a hardware approach is being adopted more often.

The IEEE 802.11g [1] wireless LAN, capable of providing data rates up to 54Mbps, has recently received increasing public attention. For the implementation of MAC layer and WEP [2] encryption/decryption, the main challenge is to find a suitable architecture meeting the Channel-speed requirement without performance loss. From the architecture point of view, there are two approaches for MAC implementation. One is firmware/hardware co-design approach, another is hardware-based approach.

The firmware/hardware co-design approach[3][4][5][6] can achieve the flexibility of design and shorten the time-to-market, but it is not a cost-effect design for the low cost and wire-speed MAC. Because it needs vast capacity of SRAM, Flash RAM memory for embedded processor operation.

The hardware-based approach [7] can achieve the low cost and wire-speed design. Because it does not need vast capacity of SRAM, flash RAM memory for embedded processor operation. After the standardization of IEEE 802.11g in 2003, the MAC remains unchanged and employs a carrier sense multiple access with collision avoidance (CSMA/CA) as fundamental access. So the main control function of hardware-based MAC can be designed and implemented using Specification and Description Language (SDL).

In this thesis, we proposed a hardware-based MAC IP that is compatible with 802.11 standard, and a hardware-based WEP IP. They both can be easily integrated into a SoC chip, while retaining the throughput at wire-speed when the WEP function is turned on. About the simulation environment, a pseudo model of 802.11 MAC is designed to generate and check 802.11 frames. For the baseband interface signals, we use the Intersil HFA3861B [8] and HFA3683 [9] as the physical layer.

WEP uses of 40-bit RC4 as its encryption mechanism. There are mainly two steps in RC4 implementation. One is the key-scheduling algorithm (KSA) and the other is the pseudo-random generation part (PRGA). KSA turns a random key into an initial permutation S. It will consume 1536 system clock. PRGA uses the permutation to generate the pseudo-random output sequence. In order to implement a channel-speed WEP engine. We use two methods to overcome the overhead of KSA operation. First, before transmit a packet to receiver, the BBP must send a preamble signal to receiver. So we can use this time to run KSA function to initial the content of S-BOX RAM at encryption operation. Second, when receive a packet from BBP, WEP engine must run decryption operation. We can use a FIFO to recover the KSA time.

Finally, we accomplish an 802.11 MAC that support DCF machines and wire-speed WEP security machines. All designs are accomplished by the Verilog RTL language. It is fully synthesizable and its operation frequency can scale up to 44MHz at the Xilinx FPGA (v2000efg1156-8) [10] device.

The organization of this thesis is described as follows. In chapter 2, the overall operation of IEEE 802.11 WLAN is discussed. The implementation issue and job partitioning process are described in Chapter 3. The function of our design is

verified in Chapter 4. The result of FPGA synthesis is presented in Chapter 5. Finally, the conclusion is given in Chapter 6.



Chapter 2

IEEE 802.11 MAC Protocol

In this chapter, we review the related knowledge about the implementation of 802.11 MAC function at chapter 3. In section 2.1, we introduce the system architecture of 802.11 standard: IBSS network and BSS network. In section 2.2, we introduce the mandatory access protocol in 802.11 standard: DCF (Distributed Coordination Function). In section 2.3, we introduce the security protocol in 802.11 standard: WEP (Wired Equivalent Privacy).

2.1 802.11 Wireless LAN Introduction

In 1997 the IEEE adopted the first wireless local area network (WLAN) standard, IEEE 802.11. This standard defines the media access control (MAC) and physical (PHY) layers for a LAN with wireless connectivity. Its maximum throughput up to 2Mbps, its spectrum is 2.4GHZ.

In 1999 the IEEE adopted the second WLAN standard, IEEE 802.11b and IEEE 802.11a. Because the standards use different portions of the spectrum, 802.11b throughput up to 11Mbps, its spectrum is 2.4GHz. 802.11a throughput up to 54Mbps, its spectrum is 5.0GHz. However, 802.11a and 802.11b are not compatible.

In 2003 the IEEE adopted the third WLAN standard, IEEE 802.11g, while maintaining backward compatibility with 802.11b products. Its maximum throughput is up to 54Mbps.

IEEE 802.11 MAC standard specifies two different machines to access data: Distributed Coordination Function (DCF) and Point Coordination Function (PCF). DCF is the basic access scheme and PCF is an optional access scheme.

DCF is a contention-based medium access scheme. It is based on a distributed random access algorithm, known as Carrier-Sense Multiple Access with Collision

Avoidance (CSMA/CA). Before attempting to transmit, sender checks whether the channel is free. If the channel is BUSY, sender defer to each other and use an exponential back off algorithm to avoid collisions.

This method is used in either Ad Hoc network or in Infrastructure network.

PCF is a contention-free based medium access scheme. Point coordinator polling all stations that list on its associated table. Associated stations can transmit data only when they are allowed to do so by the point coordinator. So PCF can provide a limited Quality Of Service function.

But this method just only is used in Infrastructure network.

2.1.1 802.11 Wireless LAN Architecture

The 802.11 architecture contains several main components: station (STA), access point (AP), independent basic service set (IBSS), basic service set (BSS), distribution system (DS), and extended service set (ESS). The wireless STA contains an adapter card, PC Card, or an embedded device to provide wireless connectivity. The AP functions as a bridge between the wireless STAs and the existing network backbone for network access.

2.1.1.1 IBSS (Independent Basic Service Set)

IBSS mode, also called ad-hoc mode, is designed for point-to-point connections. Its architecture is shown in Figure 2-1. No access points are used, and no distribution system is present. Only stations that have the same BSSID will process broadcast and multicasts.

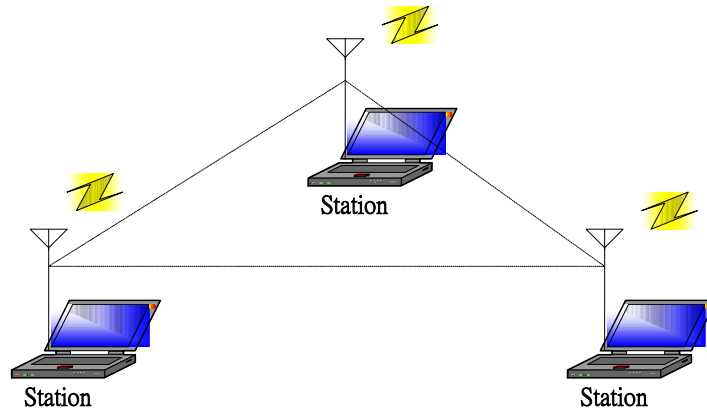


Figure 2-1 IBSS Mode

2.1.1.2 BSS (Basic Service Set)

BSS mode is the mode that typically is used. BSS mode is also called infrastructure mode. See Figure 2-2. In this mode, a number of wireless access points are connected to a wired network. Each wireless network has its own name. This name is called the SSID of the network. Access point (AP) can distribute the channel among the station. So AP also can provide a limited QoS function for station.

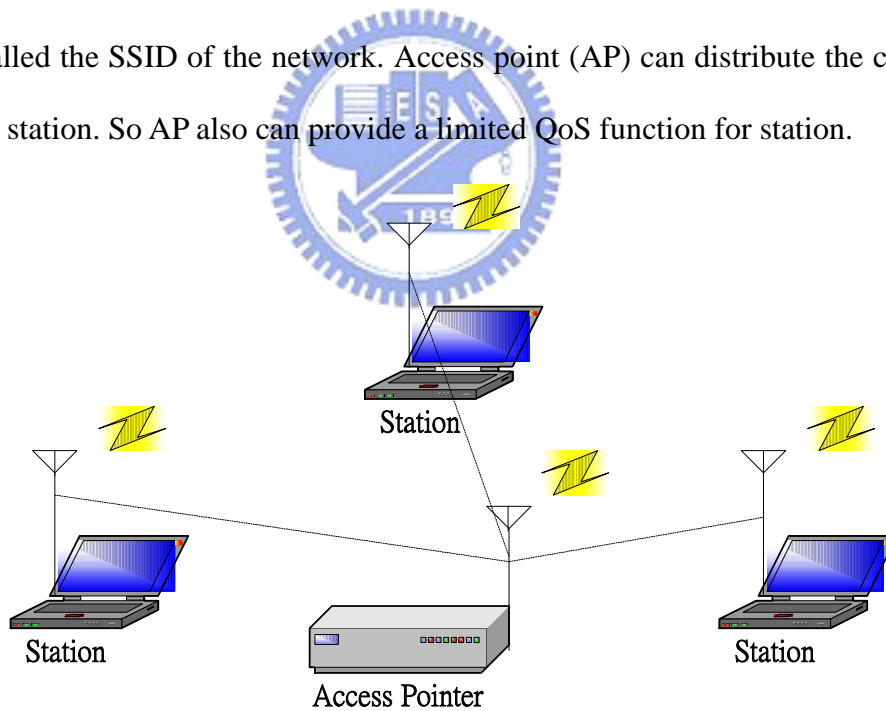


Figure 2-2 BSS Mode

2.1.2 802.11 Wireless LAN Standard

The task groups of the 802.11 standard are show as Table 2-1.

The details of some task groups are listed below.

Standard	Status	Description
802.11	Work completed	Date rate is 1 Mbps, 2Mbps Frequency band is 2.4 GHz
802.11a	Work completed	Published in 1999 Data rate up to 54 Mbps Frequency band is 5 GHz
802.11b	Work completed	Published in 1999 Date rate is 5.5 Mbps, 11Mbps Frequency band is 2.4 GHz
802.11e	Ongoing	Enhance the 802.11 MAC to increase the quality of service possible (QoS).
802.11f	Ongoing	Developing recommended practices for implementing the 802.11 concepts of Access Points and Distribution Systems. The purpose is to increase compatibility between Access Point devices from different vendors.
802.11g	Work completed	Date rate up to 54 Mbps Frequency band is 2.4 GHz
802.11i	Ongoing	Enhancing the security and authentication mechanisms of the 802.11 standard

Table 2-1 802.11 Working Groups

802.11, 802.11a, 802.11b, 802.11g : These standards are to develop a medium access control (MAC) and physical layer (PHY) specification for wireless connectivity.

802.11e: Supplementary to the MAC layer to provide QOS support for LAN applications. The purpose is to provide classes of service with managed levels of QOS

for data, voice and video applications. The proposed enhancement to DCF - Enhanced Distribution Coordination Function (EDCF) - introduces the concept of traffic categories. Each station has eight traffic categories, or priority levels. Thus stations with lower-priority traffic must wait longer than those with high-priority traffic before trying to access the medium.

802.11i: 802.11i has identified the weaknesses of the existing WEP standard. And it proposes three machines to enhance the security of MAC level.

2.2 Distributed Coordinated Function (DCF)

DCF is mandatory and based on the CSMA/CA (carrier sense multiple access with collision avoidance) protocol. With DCF, 802.11 stations contend for access and attempt to send frames when there is no other station transmitting. If another station is sending a frame, stations are polite and wait until the channel is free.

2.2.1 MAC Framing

DATA Frame

Figure 2-3 shows a generic data frame. It includes 30 bytes of frame header, 0-2312 bytes of data payload and 4 bytes of CRC-32 checksum code. Depending on the different type of data frame, some of the fields in the figure may not be used.

Control Frame

Figure 2-4 shows a RTS frame format. RA indicates destination station network address; TA indicates the source station network address. The number of microseconds required for the transmission is calculated and placed in the Duration field. The duration time = frame time + CTS frame time + ACK frame time + 3xSIFS time.

Figure 2-5 shows a CTS frame format. MAC copies the TA of RTS frame into the RA of CTS frame. The duration time = RTS – CTS time – 1xSIFS time.

Figure 2- 6 shows an ACK frame format. The RA is copied from the address 2 field of the frame being acknowledged.

Management Frame

Figure 2-7 shows a Beacon frame format. Beacon frame to announce its presence and relay information, such as timestamp, SSID, and other parameters regarding the access point to radio stations that are within range.

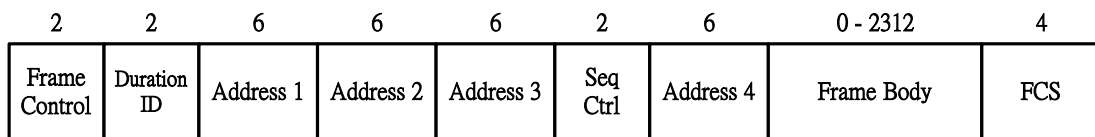


Figure 2-3 802.11 Frame Format

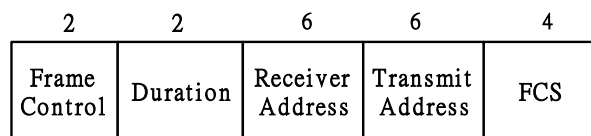


Figure 2-4 RTS Frame Format

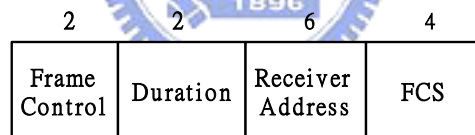
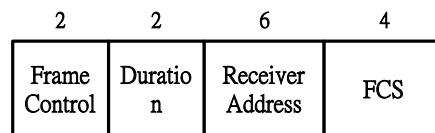


Figure 2-5 CTS Frame Format



ACK Frame

Figure 2-6 ACK Frame Format

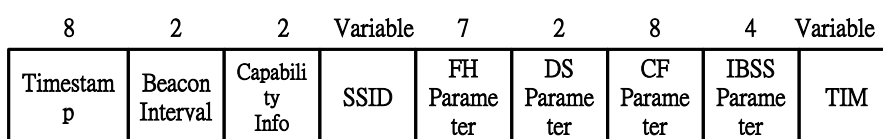


Figure 2-7 Beacon Frame Format

2.2.2 CSMA/CA+ACK Mechanism

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is the MAC protocol that is used by systems that conform to the various flavors of the IEEE 802.11 LAN Standard.

CSMA (Carrier Sense Multiple Access)

The physical layer uses a clear channel assessment (CCA) algorithm to determine if the channel is clear. Measuring the RF energy at the antenna and determining the strength of the received signal accomplish this. This measured signal is commonly known as RSSI. If the received signal strength is below a specified threshold the channel is declared clear and the MAC layer is given the clear channel status for data transmission. If the RF energy is above the threshold, the channel is declared busy and the MAC layer is given the busy channel status for data transmission.



CA (Collision Avoidance).

If the medium is free for the duration of an Inter-Frame Space (IFS), the station can start sending (IFS depends on packet type). If the medium is busy, the station has to wait for free IFS, and then the station must additionally wait a random back-off time. If another station occupies the medium during the back-off time of the station, the back-off timer stops.

IFS (Inter-Frame Space) have four different interval time: SIFS, PIFS, DIFS and EIFS.

SIFS (Short IFS) ; its duration is 10 us. highest priority, for ACK, CTS, polling response ◦

PIFS(PCF IFS) ; medium priority, for time-bounded service using PCF. $PIFS_{Time} = SIFS_{Time} + SlotTime$ ◦

DIFS(DCF IFS) ; lowest priority, for asynchronous data service. $DIFSTime = SIFS\ time + 2 \times Slot\ time$ ◦

EIFS (Extended IFS) : Extend frame Interval ◦ $EIFSTime = SIFS\ time + 8 \times ACK\ Size + Preamble\ length + PLCPHeader\ length + DIFS$ ◦

ACK (Acknowledgement)

At the end of every MPDU packet the receiver, if it has successfully received the packet, will return an ACK packet (if not received or received with errors the receiver will NOT respond). If sender didn't receive the ACK packet from receiver, the sender must retransmit previous MPDU packet.

2.2.3 RTS/CTS Mechanism

The RTS/CTS handshaking provides positive control over the use of the shared medium. The primary reason for implementing RTS/CTS is to minimize collisions among hidden stations. This occurs when users and access points are spread out throughout the facility and you're finding a relatively high number of retransmissions occurring on the wireless LAN.

RTS/CTS Machine uses a small size of RTS packet to test the channel status before MAC wants to transmit a large size of MPDU packet. If the channel is IDLE, the duration field of RTS packet will setup the NAV (Net Allocation Vector) timer that all active station in BSS.

Station can't transmit any packet before NAV timer comes down to zero. This machine can grantee that sender can send a full packet during the NAV time, so this packet didn't be destroyed by interference from another station.

Figure2- 8 shows how the NAV protects the sequence from interruption.

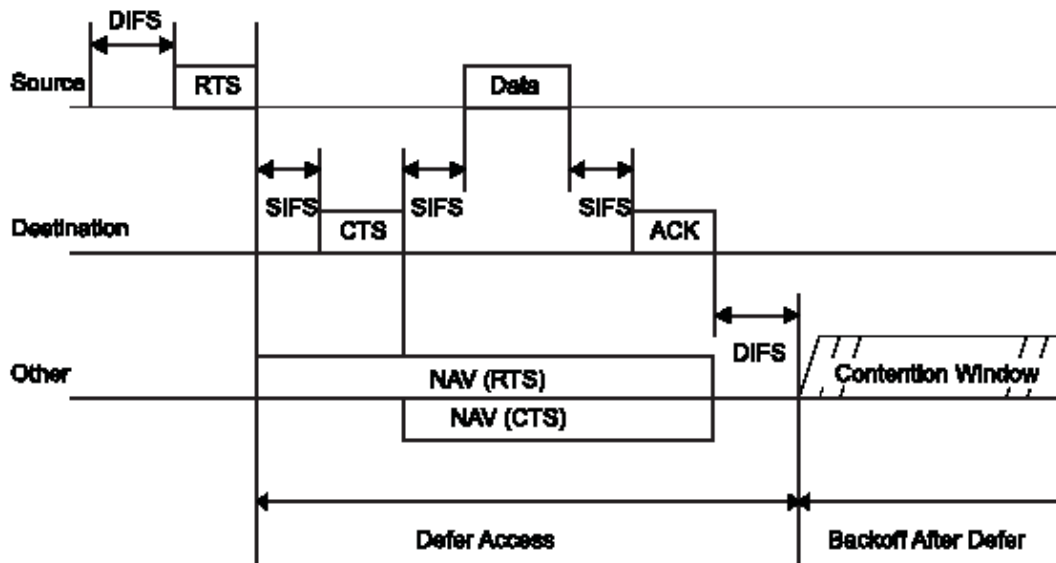


Figure2- 8 NAV Timing

2.2.4 Synchronization

In infrastructure network, the access point maintains a global TSF timer. Associated station maintain local TSF timer. The TSF timer is a 64-bit counter. So the maximum range is 2^{64} us. Access point will transmit a beacon frame that includes TSF timer to station periodically. Stations associated with an access point accept the global TSF timing value in any received Beacons, but station may add delay to the received timing value to compensate for local processing by the antenna and transceiver.

In IBSS network, all stations in the IBSS prepare to transmit a beacon frame at the Target Beacon Transmission Time (TBTT). After the TBTT interval, all stations begin to count the back off timer down to 0. If a beacon is received before the station transmission time, the pending Beacon transmission is canceled.

2.2.5 Random Back-Off Mechanism

CSMA/CA uses a back-off timer that ensures fairness. Each station starts a random back-off timer when waiting for the Contention Window (CW). This timer ticks down to zero while waiting in the contention window. Each station gets a new random timer when it wants to transmit. This timer isn't reset until the packet has

transmitted

Contention window sizes are 1 less than a power of 2-slot time. Each time the retry counter increases, the contention window move to the next greatest of two. When the contention window reaches its maximum size, it remains there until it can be reset. Figure 2-9 illustrates the growth of the contention window as the number of transmission increases, using the numbers from DSSS physical layer.

The algorithm of back off machine is shown in table 2-2.

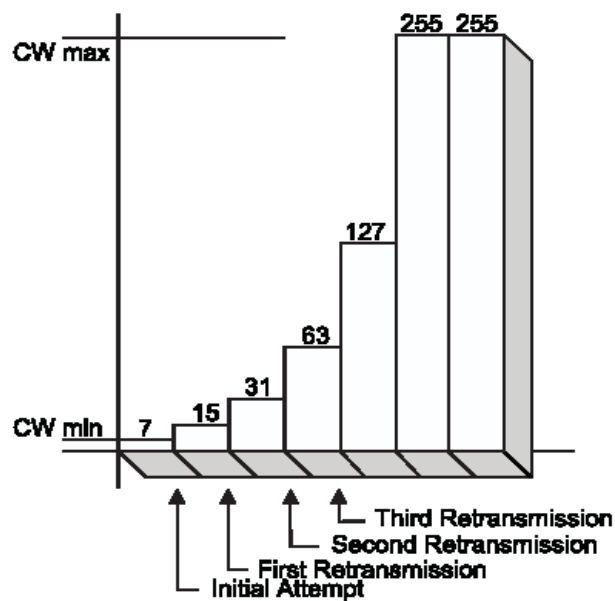


Figure 2- 9 DSSS contention window size

Back off Time= $\text{INT} (\text{CW} \times \text{Random} ()) \times \text{Slot Time}$
INT (x) is an integrity function.
CW is integrity between CW_{min} and CW_{max} .
Random() is a random number generator, its value is less than 1 ,
Slot Time = 20 us.

Table 2-2 Back off Algorithm

2.3 802.11 Security Mechanism

Because the radio broadcasts over air, anybody can receive your signal with a wireless LAN card which channel is the same as your LAN card. So the IEEE 802.11 standard defines the Wired Equivalent Privacy, or WEP, encapsulation of 802.11 data frames. The goal of WEP is intended to provide security that is equivalent to the security of a wired network.

2.3.1 Wired Equivalent Privacy Algorithm (WEP)

WEP requires the use of the RC4 stream cipher [11], which is a symmetric stream cipher. A stream cipher uses a stream of bits, called the key stream. WEP specifies the use of a 40-bit secret key. This secret key is combined with a 24-bit initialization vector (IV) to create a 64-bit RC4 key stream.

The RC4 algorithm works in two phases, Key Schedule Algorithm (KSA) and Pseudo Random Generate Algorithm (PRGA). KSA uses a variable length of key from 1 to 256 bytes to initialize a 256-byte S-BOX. The S-BOX is used for subsequent generation of pseudo-random bytes and then to generate a pseudo-random key stream.

Cipher text ($c_1 c_2 c_3$) is XOR of plaintext ($m_1 m_2 m_3$) and key streams

($k_1 k_2 k_3$). That is

$$M = m_1, m_2, m_3 \dots$$

$$K = k_1, k_2, k_3$$

$$C = c_1, c_2, c_3$$

Where $c_i = m_i \text{ XOR } k_i$

Each m_i is typically a byte (8 bits). The decryptor recovers the plaintext stream from the ciphertext stream by XORing each ciphertext bit with the corresponding key stream bit .

Table2-3 lists the Pseudo code of WEP KSA. Table 2-4 lists the Pseudo code of PRGA.


<p>KSA (Key)</p> <p>Initialization Phase:</p> <p>For I = 0 to N-1</p> <p>S [I] = I;</p> <p>J = 0;</p> <p>Scrambling Phase:</p> <p>For I = 0 to N-1</p> <p>J = (J + S [I] + Key [I mod L]) mod N;</p> <p>Swap (S [I], S [J]);</p>	
--	--

Table 2-3 WEP KSA Pseudo code

In the initialization phase of Table 2-3, it fills the array S [] with the bytes of the key. Repeat if necessary to fill the entire 256-byte array. In the scrambling phase, it starts the scrambling process that creates the pseudo random S array from the previously seeded S array. Finally, a swap function is performed to swap the value held in S [I] with the value held in S [J].

PRGA (S_n)

Initialization Phase:

$I = 0;$

$J = 0;$

Generation loop:

$I = (I + 1) \bmod N;$

$J = (J + S [I]) \bmod N;$

Swap ($S [I], S [J]$);

Output $S [(S [I] + S [J]) \bmod N];$

Table 2-4 WEP PRGAPseudo code

The integrity check algorithm, generating a CRC-32 called an integrity check value (ICV). The ICV protects the contents against tampering by ensuring that the frame has not changed in transit. The formula of CRC-32 is

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

Chapter 3

Implementation

As the complexity of ASIC design increases rapidly, the methodology of how to design correctly and quickly is getting more and more important. Using a good design flow capable of managing the possible critical issues prevents us from wasting time repeatedly. As shown in Figure 3-1, the design flow consists of two paths of design abstractions, and each path corresponds to several design steps.

(1) Verification Path:

1.Simulation Model

We begin with the 802.11 standard and system requirement to design the behavior model. The model can generate the frame of 802.11 to verify the function of WMAC design and check the receiving frame from WMAC.

2.Test Plan

The Test plan detail lists all of testing item. We follow there items to create testing patterns to verify the function of design.

(2) Design Path:

1.Module Partition

After the architecture is determined from architecture design, we proceed to module partition. At this stage, the building blocks in the predetermined architecture will be decomposed into smaller modules.

2.RTL Design

At this stage, the verilog hardware description language (HDL) RTL is used to represent this small module. The RTL codes, together with test bench, are simulated through the Cadance Verilog-XL simulator.

After accomplish above process, we create a testing environment for whole chip

verification. Simulator executes each testing item in the test plan. If all of testing item is passed, we synthesizing the RTL code in Xilinx FRGA with Foundation V3.1 FPGA Express, and give clock rate and operation condition constrains. After that, we get the gate level netlist. In FPGA P&R stage, we use Foundation V3.1 Design Manager to automatically place and route with gate level netlist and perform the STA for timing analysis.

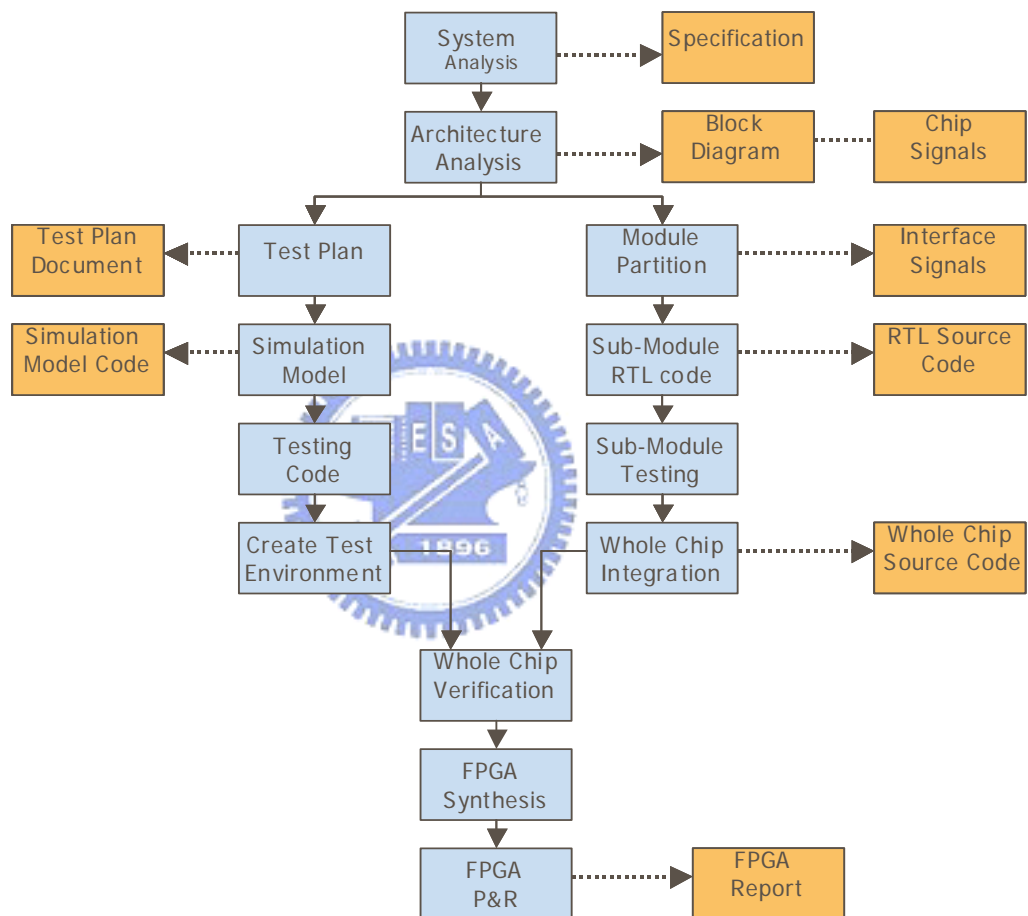


Figure 3- 1 Implementation Flow Chart

3.1 Design issues and Solutions

In this chapter, we implement a hardware-based MAC IP that is compatible with 802.11 DCF standard, and a Channel-speed WEP IP.

Hardware-based MAC

The firmware/hardware co-design approach is not a cost-effective design for the low cost and Channel-speed MAC. Because it needs vast capacity of SRAM, Flash RAM memory for embedded processor operation. After the standardization of IEEE 802.11g in 2003, the MAC remains unchanged and employs a carrier sense multiple access with collision avoidance (CSMA/CA) as fundamental access. So the main control function of hardware-based MAC can be designed and implemented using hardware architecture. The following is the solution of the design techniques used in hardware-based MAC to meet the 802.11 MAC layer function:

1. The main control function of hardware-based MAC can be designed and implemented using Specification and Description Language (SDL) of 802.11 DCF standard.
2. Use an embedded beacon SSRAM to store beacon packets and designed a beacon control circuit to load the data from beacon SSRAM on the fly.
3. Auto-calculating the byte count from the LENGTH field of PLCP header between WMAC and BBP, and auto-setting the content of BBP registers using their values.

Wire-speeded WEP

The Channel-speed WEP uses 40-bit RC4 as its encryption mechanism. There are mainly two steps in RC4 implementation. One is the key-scheduling algorithm (KSA) and the other is the pseudo-random generation part (PRGA). KSA turns a random key into an initial permutation S. It will consume 1536 system clocks every packet. This time limit the throughput of 802.11 network when WEP function turns on. The following is the solution of the design techniques used in WEP engine to meet the desired performance.

1. Use a FIFO RAM to recover the KSA time of RC4 algorithm at WEP decryption.
2. Use the PLCP preamble time to recover the KSA time of RC4 algorithm at WEP

encryption.

Pseudo simulation model

For the function verification, we developed three pseudo-models to simulate the operation of physical BBP chip and RF chip.

1. Design a Pseudo simulation model that can simulate the operation of 802.11 MAC with verilog RTL code.
2. Design a Pseudo simulation model that can simulate the operation of BBP chip with verilog RTL code.
3. Design a Pseudo simulation model that can simulate the operation of RF chip with verilog RTL code.

3.2 System Architecture Design

To specify the WMAC chip from the functional and architectural point of view, to produce the detailed hardware specification of WMAC. The output from system architecture design stage will be a chip specification document outlining all the technical requirements. Having completed the document and had it agreed, this detailed analysis will take account of all or many of the following issues.

- Selecting a BBP chip and RF chip
- Pin assignment requirements
- Register set definition

Simplified whole chip architecture is illustrated in Figure 3-2, which shows that all the building blocks for MAC. The MAC is designed to implement the timing critical and processing-intensive MAC functions, such as timing synchronization, CRC generation, RTS/CTS packet, etc. it coordinated with BBP through MMI interface.

The WEP encryption/decryption is designed to implement the WEP function. The FIFO is a dual port memory device. With one port to the MAC, another to the

WEP decryption controller, MAC packets that are buffered in order to provide a time to initial the value of S-BOX before encrypting the ciphertext.

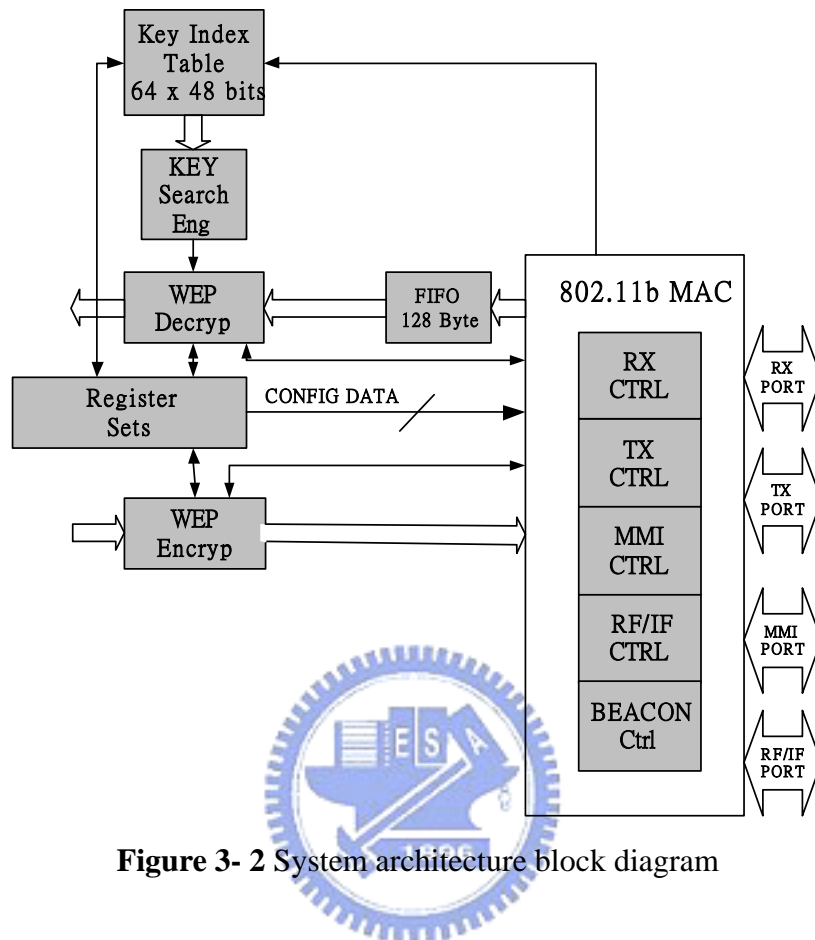


Figure 3- 2 System architecture block diagram

3.2.1 Chip Specification

The features of designed MAC and WEP engine are as flows :

- **802.11 MAC**
 - IEEE 802.11, IEEE 802.11b
 - MAC implements with State Machine
 - No External SRAM needed
 - Support Infrastructure AP, Station and IBSS under DCF
 - RTS/CTS, Beacon, ACK detection/generation
 - Support four MAC address (unicast)
 - Support MMI Interface control for Intersil BBP

- Front end chip power sequence control

■ WEP

- Internal Encryption/Decryption engine for WEP function
- Support RC4, 40/104 bits key length selectable
- ICV generation and check
- Support 64-entry Key pointer table
- Individual Key search engine

3.2.2 Chip Pin assignment

I based on Intersil HFA3861B to define BBP interface signals, and based on Intersil HFA3683 chip to design the RF interface control signals. The boundary signals of chip are shows as Appendix A. There signals are partitioned into several parts:

Group A : to RF Chip (HFA3683)

Group B : to baseband processor (HFA3861B)

Group C : to register sets

Group D : to buffer management

Group E : from buffer management

Group F : from baseband processor (HFA3861B)

Group G : from registers sets

3.2.3 Sub-module partition

Our proposed architecture for the IEEE 802.11b MAC functionality is illustrated in Figure 3-3. As shown in the figure, it includes five major module, the Receive module, Transmit module, RF Interface control module, Beacon control module and

Timing Sync Function (TSF) module. The MAC provides an embedded SSRAM for the storage of the beacon frame. Receive module handles the frame analysis, control the handshake between the MAC and Buffer Manager Interface.

Transmit module generate the CTA and ACK frames in response to RTS and Data frame. Is also generate the RTS frame when the length of MPDU is longer than RTS-through.

Timing Sync Function module maintain a 64-bits width timer. Beacon control module controls a 64x16 SSRAM to store beacon frame for transmission. RFIF interface module can access the registers of RF chip. Detail description on the functionality of there modules is listed below.

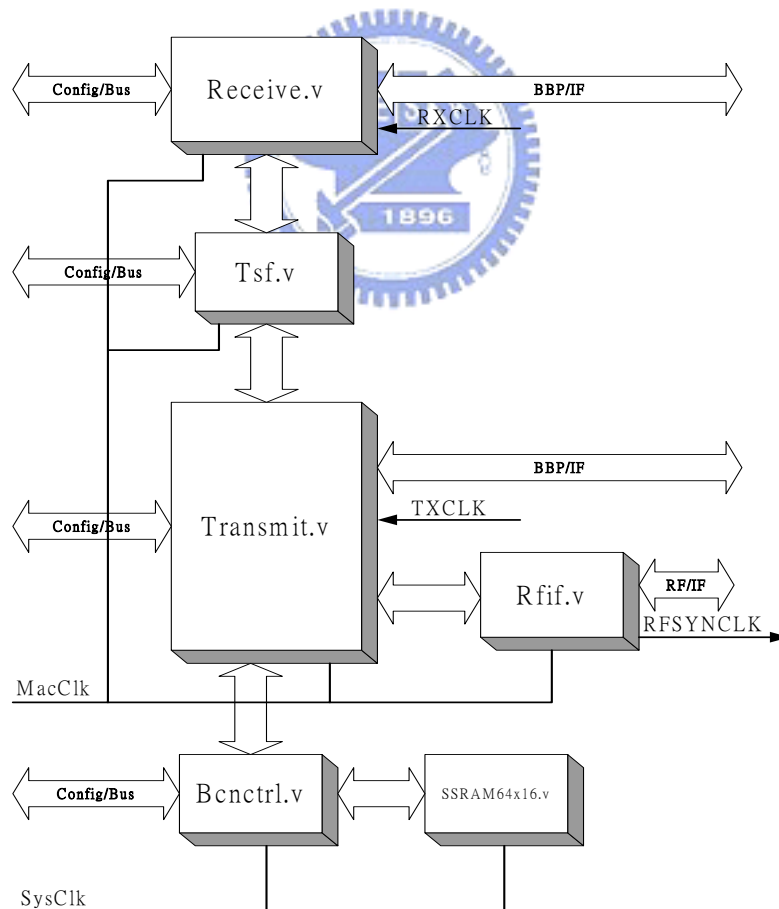


Figure 3- 3 MAC Function Block Diagram

□ **Sub module hierarchical**

First step, In order to reduce the complexity of design, The WMAC will be partitioned into twenty-two sub-modules for implementation. The relation of sub-module is show in Figure 3-4.

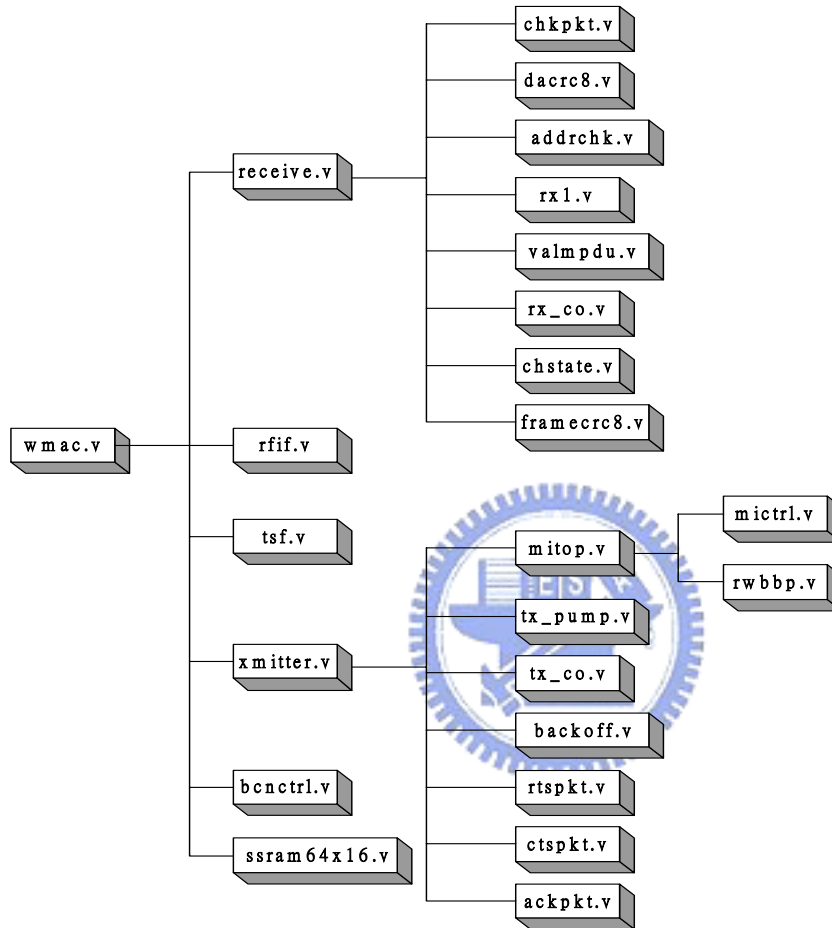


Figure 3- 4 Sub module hierarchical diagram

3.3 DCF Implementation

The implementation of a DCF mechanism is based on two process: transmit process and the receive process, each of which executes a number of functions to manage the outgoing or incoming frames. When we analyze the FSM of MAC on the basis of this IEEE 802.11 SDL system, we can notice that the hardware logic block should be generally limited in timing critical and processing-intensive functions.

In these features, protocol control function for transmission and reception, CRC check, FCS check, Octet data transfer, CCA check, RTS/CTS/ACK frame, and back off mechanism are possible for the implement as hardware using Verilog.

3.3.1 Receive Module

Receive module's pin as show in figure 3-5.

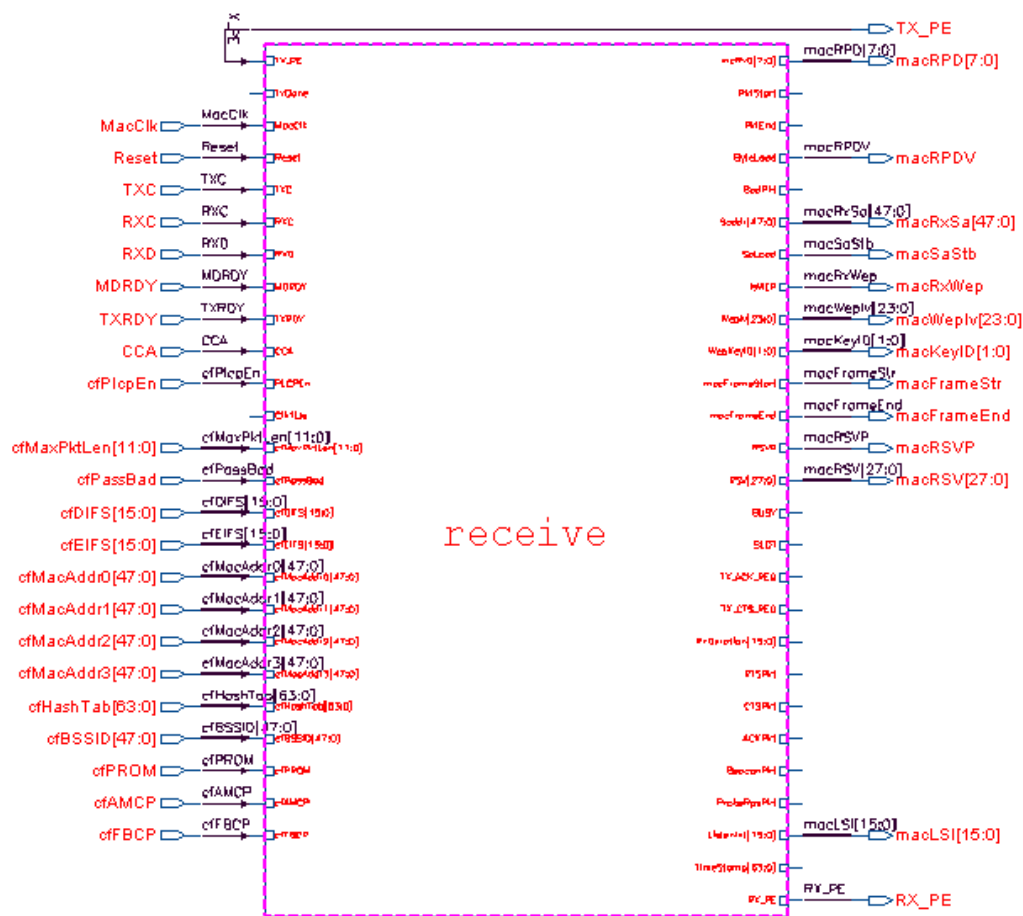


Figure 3- 5 receive Module Interface

This module is composed of eight sub-module, chkpkt, dcrc8, addrchk, rx1, valmpdu, chstate, rx_co, and framecrc8.

Chkpkt module parses the contents of 802.11 frames. Dcrc8 is used to check the CRC value of received packet. addrchk module is used to filter the unicast or multicast address. Valmpdu handles the DIFS or EIFS timer. Chstate module handles the channel state, like IDLE or BUSY.

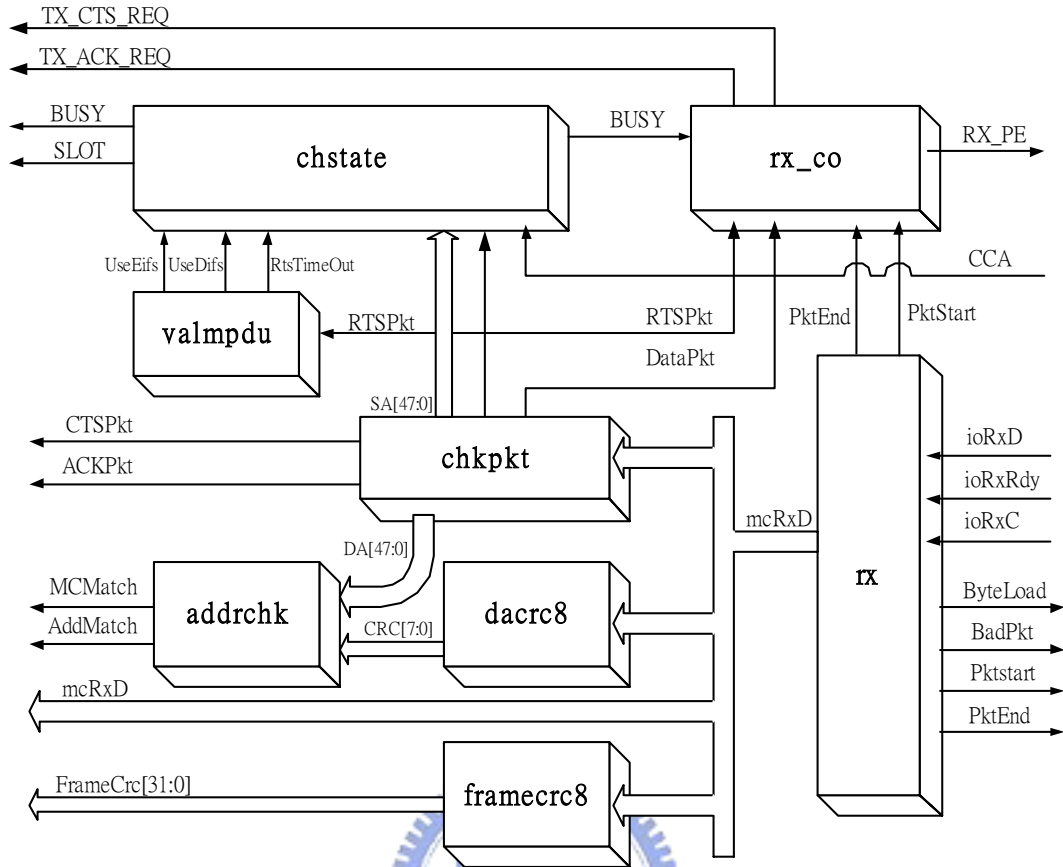


Figure 3- 6 Receive Module Function Block Diagram

3.3.1.1 chstate Module

chstate module's pin as show in figure 3-7, this module handles the channel state and NAV Timer. If BUSY signal is high, indicate the channel is BUSY. If BUSY signal is low, indicates the channel is IDLE. SLOT signal is a clock signal which period is 10us. It is used to back off module as a clock source, but it still only active on back off period. Its FSM flow chart is show in figure 3-8.

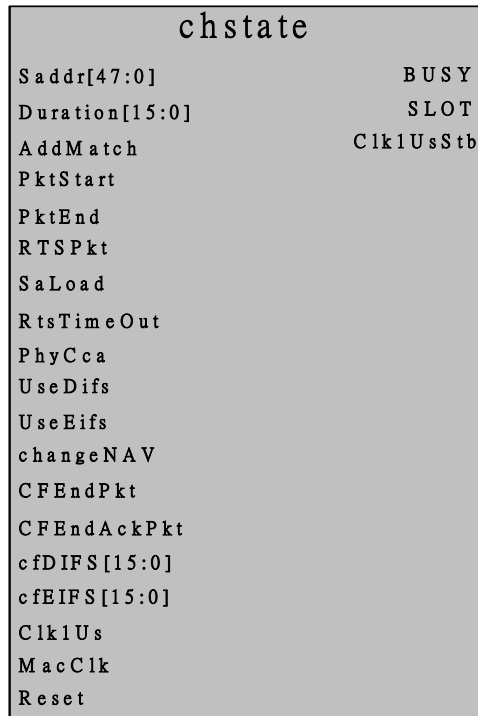


Figure 3- 7 chstate Module Interface

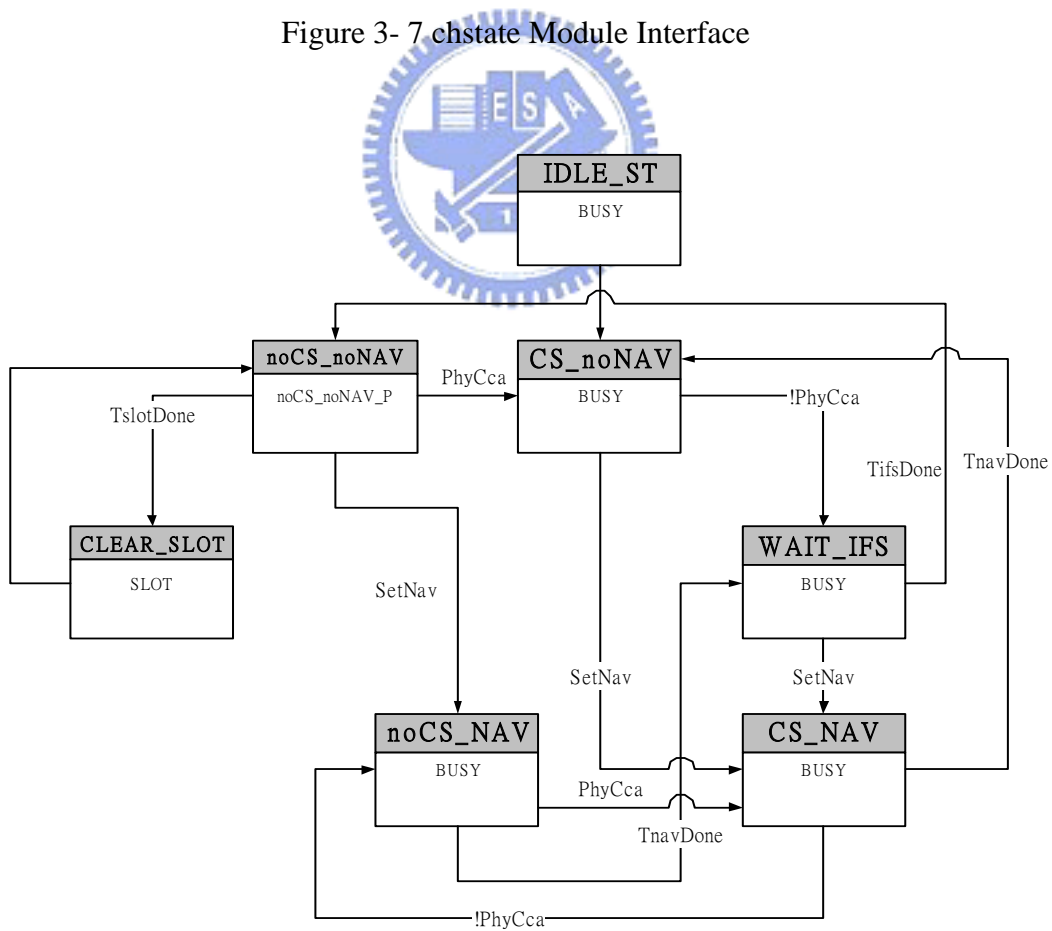


Figure 3- 8 chstate Module state machine flow chart

3.3.1.2 valmpdu Module

valmpdu module's pin assignments as show in figure 3-9. It decide the inter frame gap of response packet. When MAC receives a good packet from BBP, it asserts the UseDifs signal. When MAC receive a bad packet (CRC error, No Ack) from BBP. It assert UseEifs signal. Its FSM Flow Chart can be seen in Figure 3-10

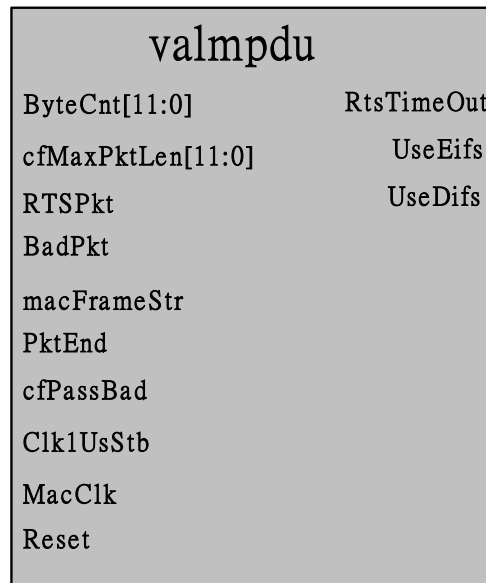


Figure 3- 9 valmpdu Module Interface

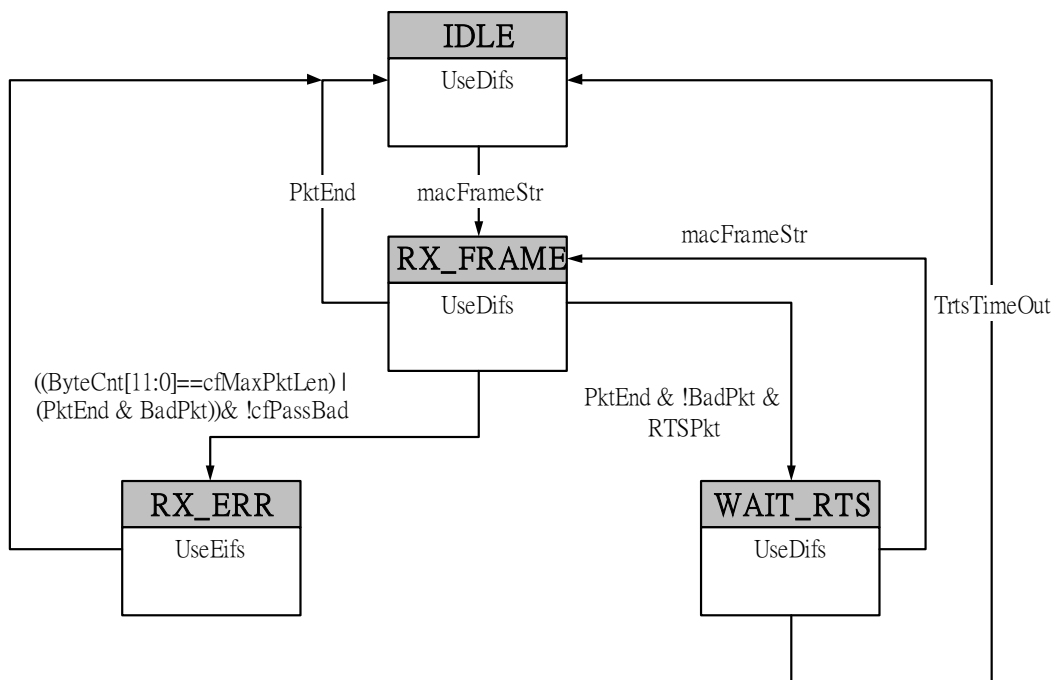


Figure 3- 10 valmpdu Module state machine flow chart

3.3.1.3 rx_co Module

Its pin assignment is shown in Figure 3-11, it use TX_ACK_REQ signal to indicate to tx_pump module when an ACK packet is need to be response for sender.

It uses TX_CTS_REQ signal to indicate to tx_pump module when a CTS packet is need to be response for sender. RX_PE signal is used to active the receiving function of BBP. RxDuration is used to update NAV timer.

The flow chart for Finite State Machine is shown in Figure 3-12. The initial state is IDLE state. When rx module report a PktEnd (packet end), if its frame type is RTS packet or DATA packet. The FSM will entry the WAIT_SIFS state and wait a SIFS time. Then FSM will entry the TX_ACK or TX_CTS state to generate TX_ACK_REQ or TX_CTS_REQ signal to transmit module.

Figure 3-12 shows the flow chart of FSM.

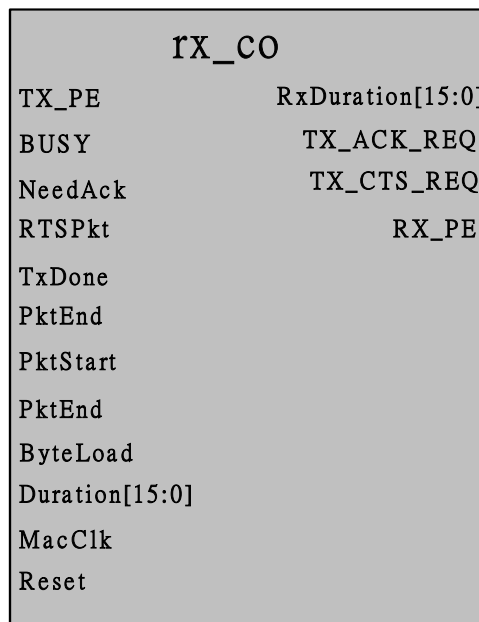


Figure 3- 11 rx_co Module Interface

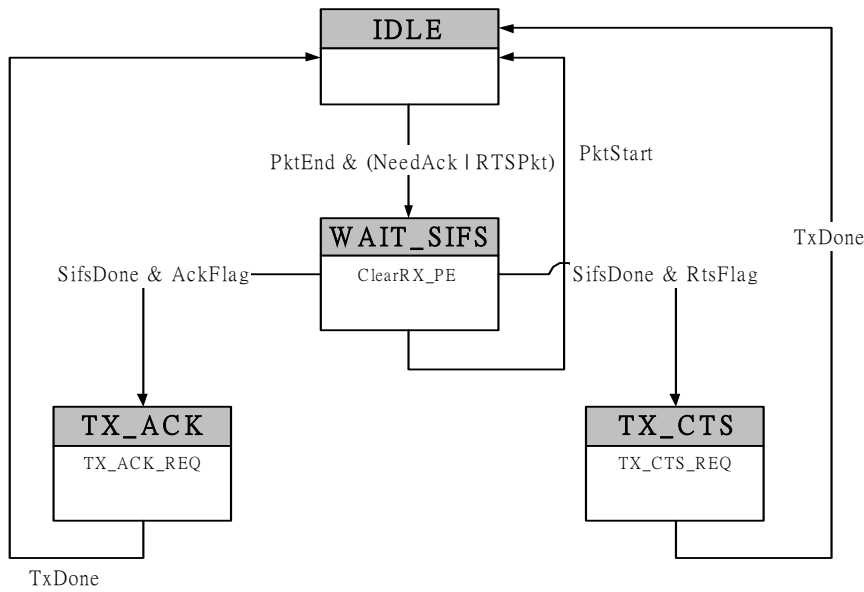


Figure 3- 12 rx_co Module state machine flow chart

3.3.1.4 chkpkt Module

Its pin assignment is shown in Figure 3-13. This module is used to identify the type of packet. Such as PS-Poll, RTS, CTS, ACK and Beacon packet. It also extracts header data from packet for another module to use, such as Destination address, Source address, Duration, Beacon interval, listen interval and Timestamp.



Figure 3- 13 chkpkt Module Interface

3.3.2 Transmit Module

Its block diagram is shown in Figure 3-14. It is composed of rtspkt.v, ctspkt.v, ackpkt.v, mitop.v, tx_pump.v, tx_co.v, and backoff.v.

Rtspkt.v provides a RTS packet for tx_pump.v to use. ctspkt.v provides a CTS packet for tx_pump.v to use. ackpkt.v provides a ACK packet for tx_pump.v to use. mitop is used to access the register of BBP chip. tx_pump.v is used to send MPDU to BBP chip. tx_co.v is used to generate request signal to tx_pump.v to active transmission function. Backoff.v is used to maintain random back off timer for DCF function.

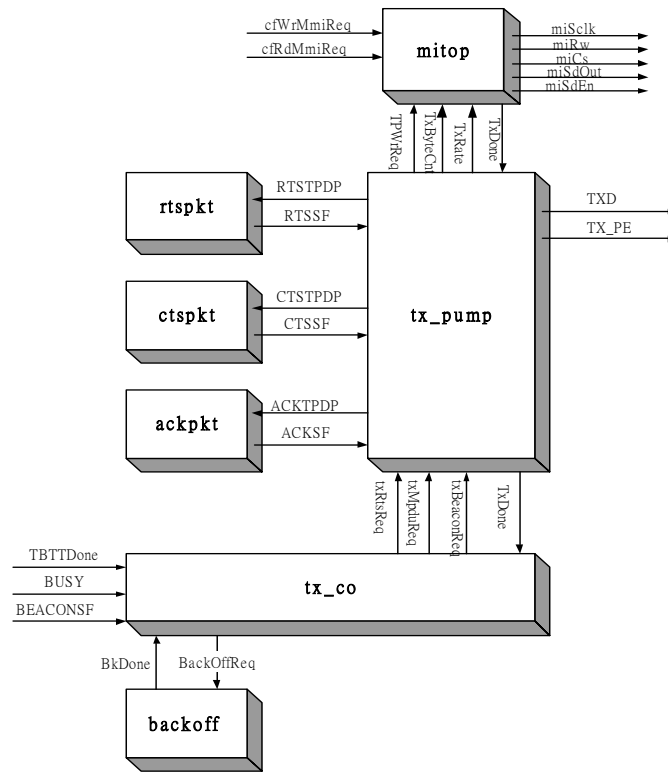


Figure 3- 14 Transmit Module Function Block Diagram

3.3.2.1 tx_co Module

Its pin assignment is shown in Figure 3-15.

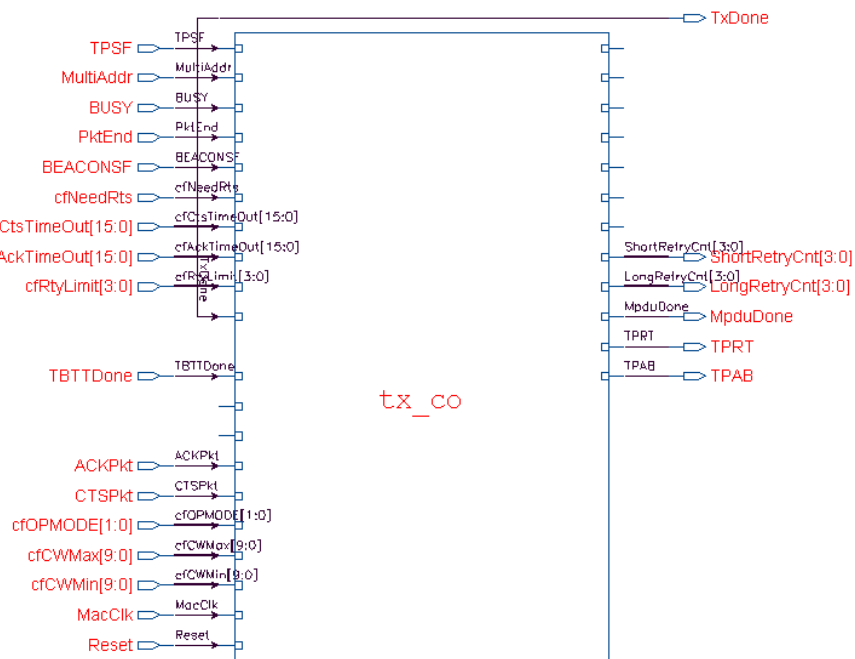


Figure 3- 15 tx_co Module Interface

Tx_co.v is used to generate request signal to tx_pump.v to active transmission function. We can control its operation through configuration register. Setting cfRtlylimit value can limit the numbers of retry. CfCtsTimeOut and cfAckTimeOut value can set the time-out timer. WMAC will generate a TPAB signal to abort current transmission when time-out timer is expired.

3.3.2.2 AckPkt Module

Its pin assignment is shown in Figure 3-16. It uses input signals to construct an ACK packet for tx_pump to use. RXADDR represents the address of receiver. It uses ACKSF to indicate to tx_pump.v when it is ready for the transmission. TPDP is used to move data from ackpkt.v to tx_pump.v through ACKDATA [7:0] bus.

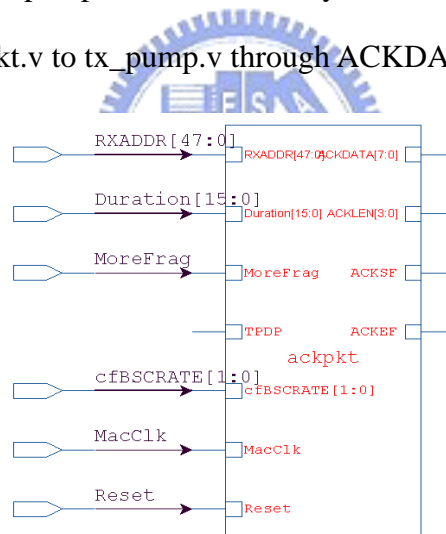


Figure 3- 16 ACKPkt Module Interface

3.3.2.3 rtspkt Module

Its pin assignment is shown in Figure 3-17. It uses input signals to construct a RTS packet for tx_pump to use. DAADDR represents the address of receiver. SAADDR represents the address of sender. It uses RTSSF to indicate to tx_pump.v when it is ready for the transmission. TPDP is used to move data from rtspkt.v to tx_pump.v through RTSDATA [7:0] bus.

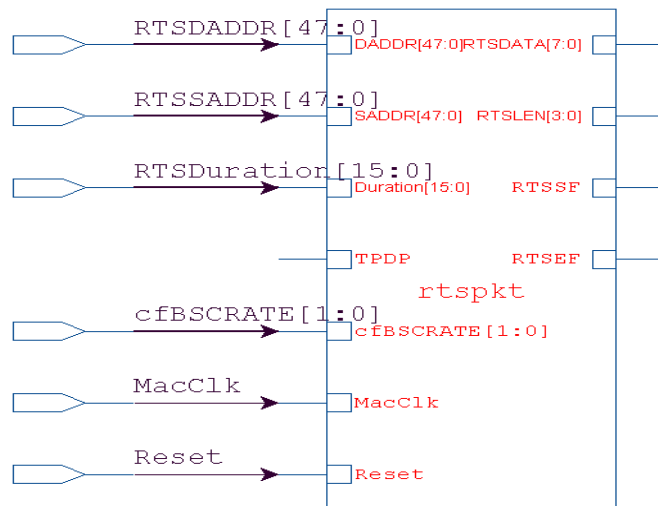


Figure 3- 17 RTSPkt Module Interface

3.3.2.4 ctspkt Module

Its pin assignment is shown in Figure 3-18. It uses input signals to construct a CTS packet for tx_pump to use. RAADDR represents the address of receiver. It uses CTSSF to indicate to tx_pump.v when it is ready for the transmission. TPDP is used to move data from ctspkt.v to tx_pump.v through CTSDATA [7:0] bus.

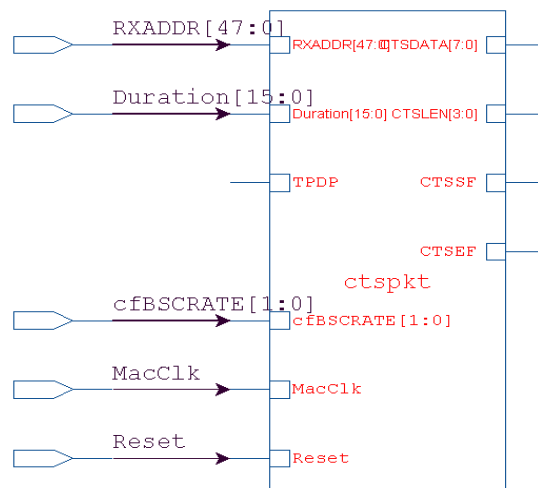


Figure 3- 18 CTSPkt Module Interface

3.3.2.5 back off Module

Its pin assignment is shown in Figure 3-19. This module performs the back off algorithm of DCF. It can generate a random back off period for an additional deferral time before transmitting.

Slot Time is a clock signal which period are 20 us.

CW(Content Window) is an integer within the range of values of BBP characteristics aCW_{min} and aCW_{max} , $aCW_{min} \leq CW \leq aCW_{max}$.

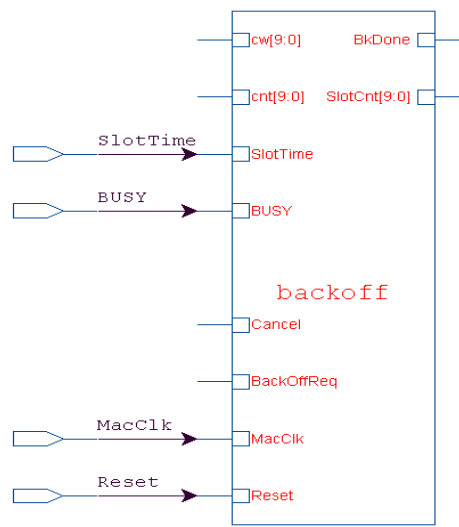


Figure 3- 19 back off Module Interface

3.3.3 Time Stamp Module

Its pin assignment is shown in Figure 3-20. This module maintains a copy of the timing synchronization function (TSF), which is a local timer synchronized with the TSF of every other station in the basic service area. The TSF is based on a 1 MHz clock. In BSS station mode. Station saves the timestamp from the beacon or probe response. The received timestamp updates the local timer only if it is later than the local timer. When the local timer reaches the Beacon Interval (BeaconInt[15:0]). TBTTDone signal will be asserted. In BSS AP mode. When the local timer reaches the Beacon Period (cfBP [15:0]). TBTTDone signal will be asserted.



Figure 3- 20 tsf Module Interface

3.3.4 Beacon Control Module

Beacon packet announces the existence of network and important information of station. It is transmitted at beacon interval and it allows other stations to find and to come to synchronization. Not all of the fields are present in all beacons. Optional fields are present only when there is a reason for them to be used. So Host CPU must fill appropriate value to there field before transmit. As a broadcast message, a beacon is not acknowledged so that the invocation of beacon generation function may be delayed due to ongoing transmission at the TBTT, subsequent beacons shall be scheduled at the next TBTT.

In this thesis, we designed an embedded beacon SSRAM to store beacon packets and designed a beacon control circuit to load the data from beacon SSRAM on the fly. This machine can reduce the overhead of moving data from Host to MAC.

Beacon control module controls the operation of reading or writing beacon packet. Its pin assignment is shown in Figure 3-21. When host want to write beacon

data into Beacon RAM. It follows below steps.

Step 1: Setting the start address of data area using cfTabAddrWrN and cfTableAddr signal.

Step 2: Load the payload data into Beacon RAM using cfTabDataWrN and cfTableAddr signal.

Step 3: When all data are stored in beacon RAM. Host will set the cfbecOwn to inform WMAC to access Beacon RAM.

Step 4: WMAC load payload data with TPDP signal when TBTTDone is asserted.

Step 5: When all data are transmitted by WMAC. WMAC will clear the bcnClrOwn to inform Host to access Beacon RAM.



Figure 3- 21 bcnctrl Module Interface

3.4 WEP Algorithm Implementation

In order to implement a wire-speeded WEP engine. We use two methods to overcome the overhead of KSA operation. First, before transmit a packet to receiver, the BBP must send a preamble signal to receiver. So we can use this time to run KSA function to initial the content of S-BOX RAM at encryption operation.

If we select a long PLCP preamble, this time is 196 us (preamble + header). If we select a short PLCP preamble, this time is 96 us (preamble+header).

The KSA procedure will consume 1536 (256 + 5x256) system clock. If the system clock is 44 MHz. So its total process time is $1536 \times 22.7 \text{ ns} = 34.867 \text{ us}$. It is less than short PLCP preamble time (96 us). In PRGA phase, the PRGA engine will consume 4-system clock to deal with one data byte. This time is $4 \times 22.7 \text{ ns} = 90.8 \text{ ns}$. It is less than 148 ns (transfer time of one data byte at 54Mbps throughput). So we can guarantee this design can come to wire-speed throughput at encryption operation.

There timing relation is shown in figure 3-22

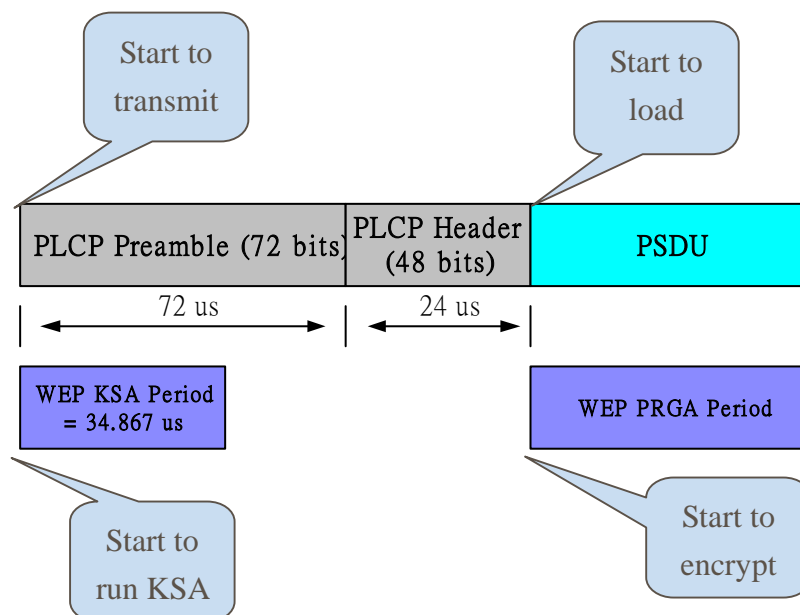


Figure 3-22 WEP Encryption timing

Second, when receive a packet from BBP, WEP engine must run decryption operation. We can use a FIFO to recover the KSA time. The KSA time is 34.867 us at 44 MHz. The one byte transmit time is $8 \times 18.5 \text{ ns} = 148 \text{ ns}$ at 802.11g (54Mbps), the FIFO depth = KSA time/ one byte transmit time. We can deduce that FIFO depth need $34867\text{ns}/148 \text{ ns} = 235.6$. So we can adopt a 256 Byte FIFO RAM to recovery the KSA time at decryption operation. Its operation flow is shown in figure 3-23.

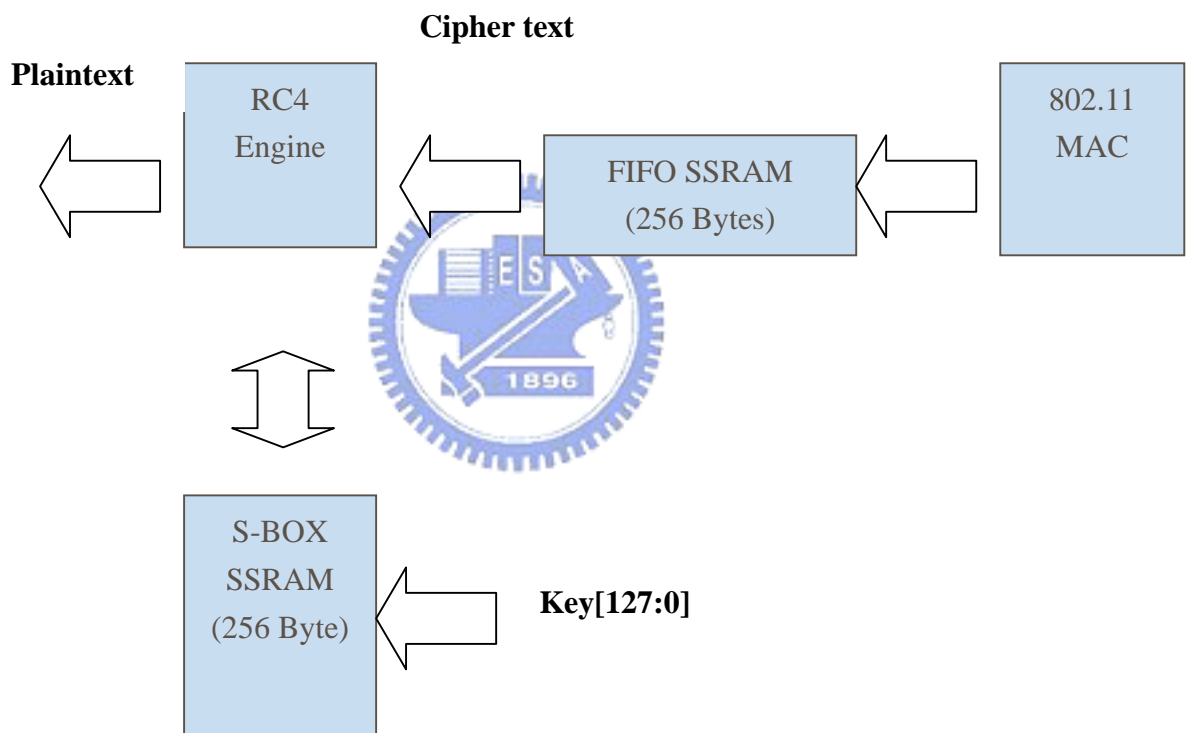


Figure 3-23 WEP decryption flow

The RC4 engine pin assignment is shown in Figure 3-24.

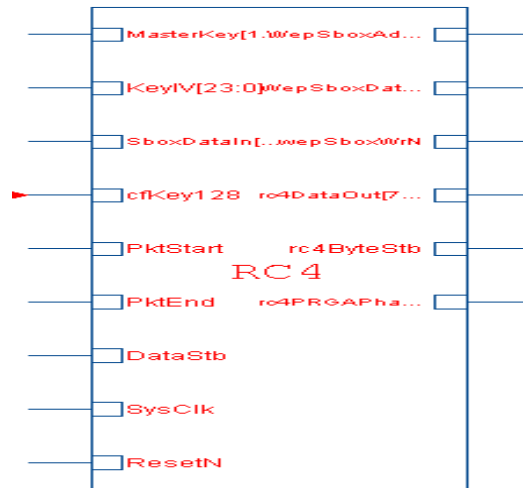


Figure 3- 24 RC4 Module Interface

Its state machine can be seen in Figure 3-25.

State "KSA_INIT" is used to initial S-Box S [I] RAM.

State "KSA_SCRAMB", "KSA_SCRAMB1", "KSA_SWAP", "KSA_SWAP0" and "KSA_SWAP1" are used to generate a pseudorandom sequence, and store their sequences into S-Box RAM

State "PRGA" to state "OUTBYTESTB" are used to generate a pseudorandom key sequence to bitwise XORed with a block of plaintext.

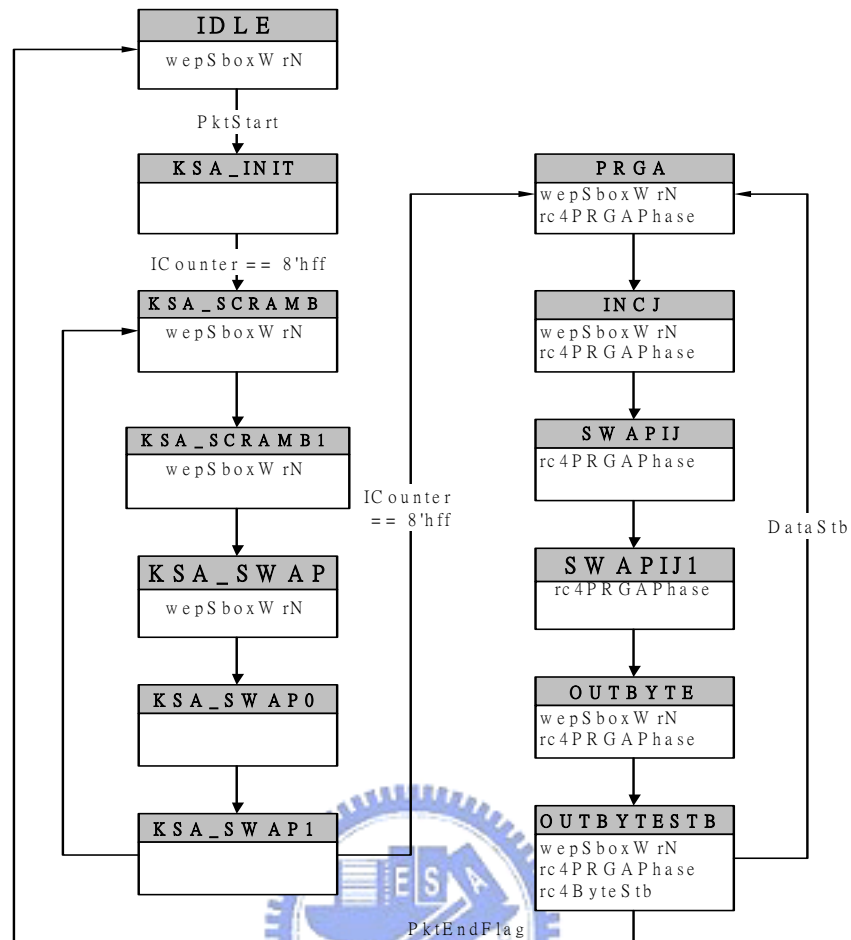


Figure 3- 25 RC4 state machine

3.5 BBP Interface Implementation

We based on Intersil HFA3861B to define BBP interface signal, this chip’s pin assignment is shown in Figure 3-26

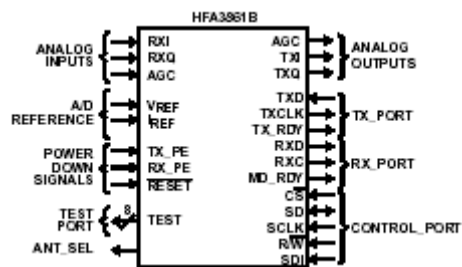


Figure 3- 26 HFA3861B Pin Assignment Diagram

WMAC can access the registers of HFA3861B through SCLK, SD, RW and CS

signals. SCLK is the clock for SD serial bus. The data on SD bus is clocked at rising edge. SD is a serial bidirectional data bus, which is used to transfer address and data to/from the internal registers. RW is an input to HFA3861B used to change the direction of the SD bus when reading or writing data on the SD bus. A high level indicates read while a low level is a write. CS is a Chip selects for the device to active the serial control port. The timing relationships of there signals are illustrated in Figure 3-27 and 3-28.

The WMAC initials the transmit sequence by asserting TX_PE. TX_PE envelops the transmit data packet on TXD. The HFA3861B responds by generating preamble and PLCP header. Before the last bit of the header is sent, the HFA3861B begins generating the TXCLK to input the serial data on TXD. TX_RDY, which is an output from HFA3861B, is used to indicate to the WMAC that the preamble has been generated and the BBP is ready to receive the data packet to be transmitted from the WMAC.

RXCLK is an output from the HFA3861B and is the clock for the serial demodulated data on RXD. MDRDY is an output from HFA3861B and it may set to go active after the SFD or CRC field. The timing relationships of there signals are illustrated in Figure 3-29 and 3-30.

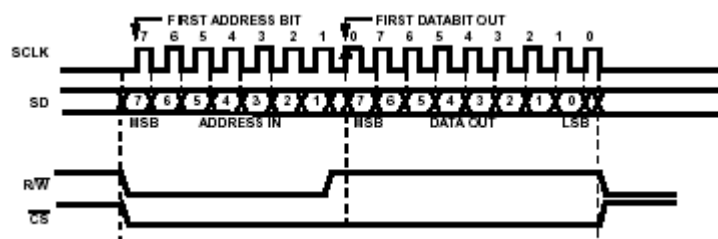


Figure 3- 27 HFA3861B control port read timing

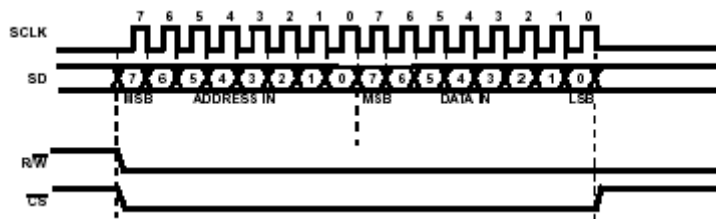


Figure 3- 28 HFA3861B control port write timing

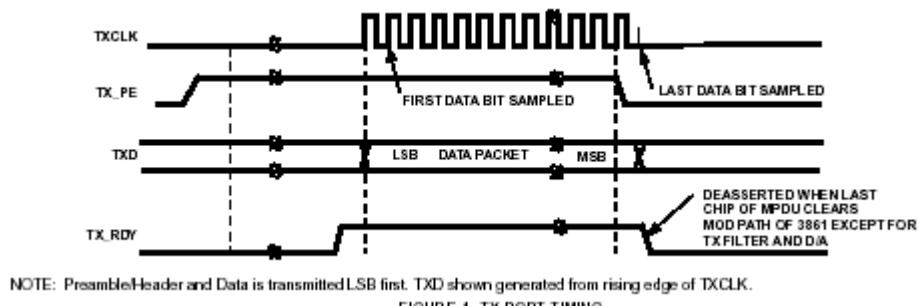


Figure 3- 29 HFA3861B TX port timing

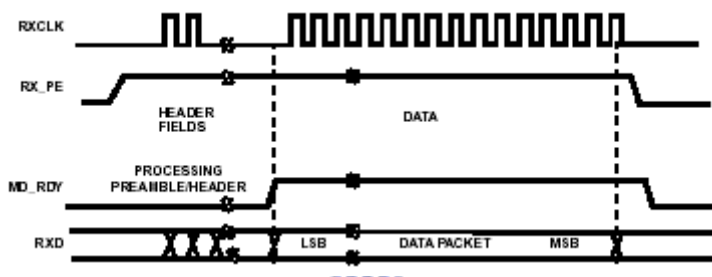


Figure 3- 30 HFA3861B RX port timing

Mitop.v module is used to control the external BBP chip. It includes two sub-module, rwbbp.v and mictrl.v. The connection relationship of their sub-module is illustrated in Figure 3-31.

Number of octets = Floor(((Length x R) / 8) – P) – Length Extension bit.

R = Data rate in Mbit/s

P = 0 for CCK, 1 for PBCC

Floor(X) = returns the largest integer value less than or equal to X.

The example of LENGTH calculations under CCK mode is show as Table 3- 1

CCK at 11Mbit/s

Tx octes	Octes (x 8/11)	LENGTH	Length extension bit	LENGTH (x 11/8)	Floor (X)	RX octets
1023	744	744	0	1023	1023	1023
1024	744.7273	745	0	1024.375	1024	1024
1025	745.4545	746	0	1025.75	1025	1025
1026	746.1818	747	1	1027.125	1027	1026

Table 3-1 Example of LENGTH calculations under CCK mode

The example of LENGTH calculations under PBCC mode is show as Table 3- 2.

PBCC at 11Mbit/s

TX octets	Octets (x 8/11)	LENGTH	Length extension bit	LENGTH (x 11/8)	Floor (X)	RX octets
1023	744.7273	745	0	1024.375	1023	1023
1024	745.4545	746	0	1025.750	1024	1024
1025	746.1818	747	1	1026.125	1026	1025

1026	746.9091	747	0	1026.125	1026	1026
------	----------	-----	---	----------	------	------

Table 3- 2 Example of LENGTH calculations under PBCC mode

Its FSM Flow Chart can be seen in Figure 3-32. It accepts two types of request signals, TPWrReq and cfWrMmiReq. TPWrReq, which is an input from host interface. It is used to indicate to the rwbbp.v module that WMAC begins to request a packet transmission. CfWrMmiReq that is an input from host interface. It is used to indicate to the rwbbp.v module that Host begins to request to access the internal register of BBP chip.

After the length of packet is calculated. Rwbpp.v initials the reading/writing request by asserting RdMmiReq or WrMmiReq signal to the mictrl.v module. Then mictrl.v responds by generating BBP chip control signals to write the internal register of BBP chip. There register is Length, Rate, and Service register.

The format of PLCP header is shown in Figure 3-33. mictrl.v Module FSM Flow Chart can be seen in Figure 3-34.

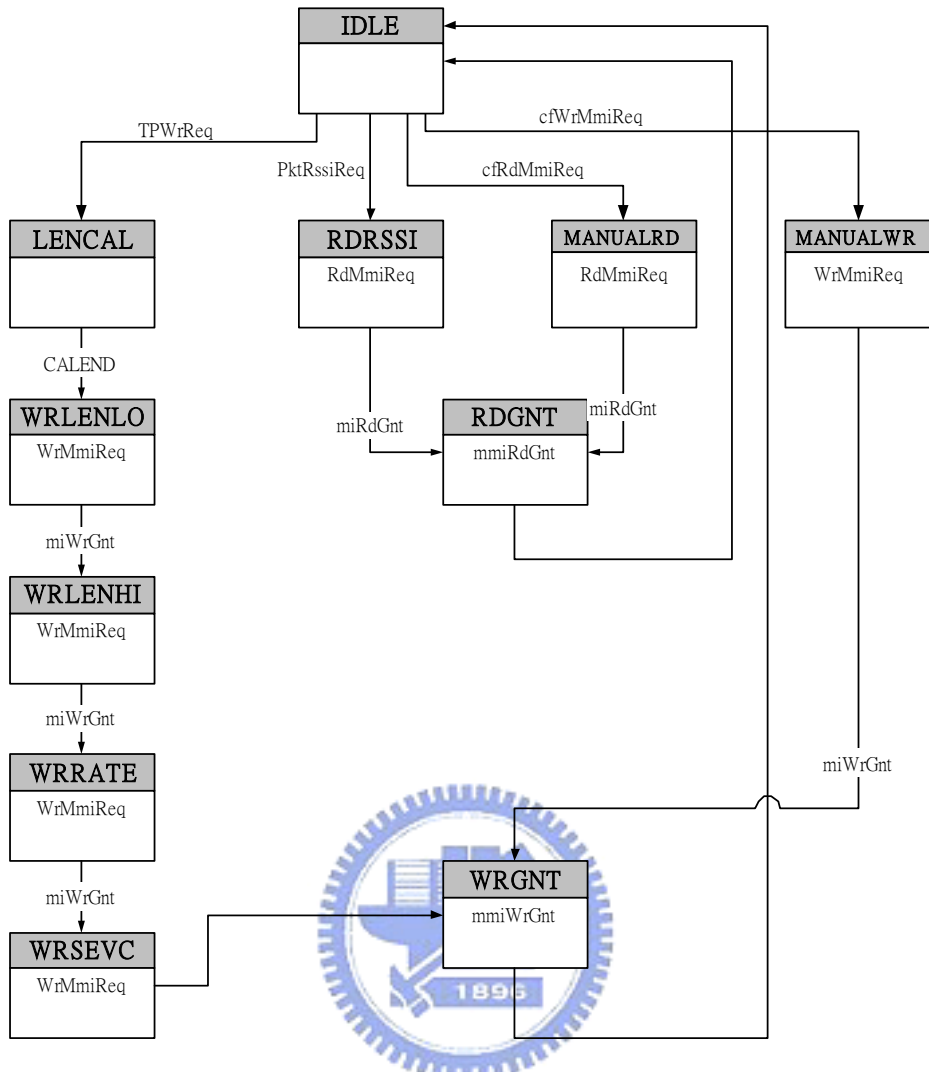


Figure 3-32 rwbbp Module FSM flow chart

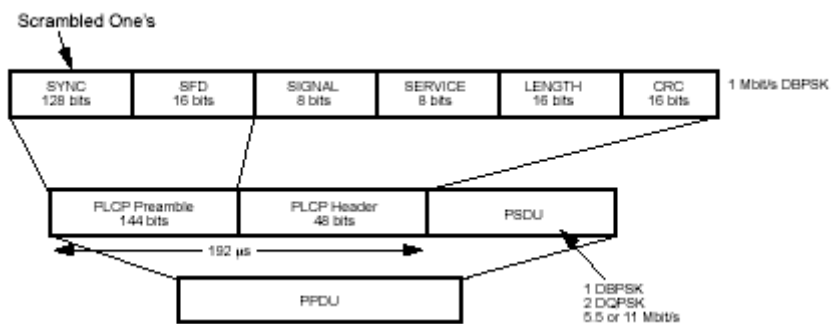


Figure 3-33 PLCP Header

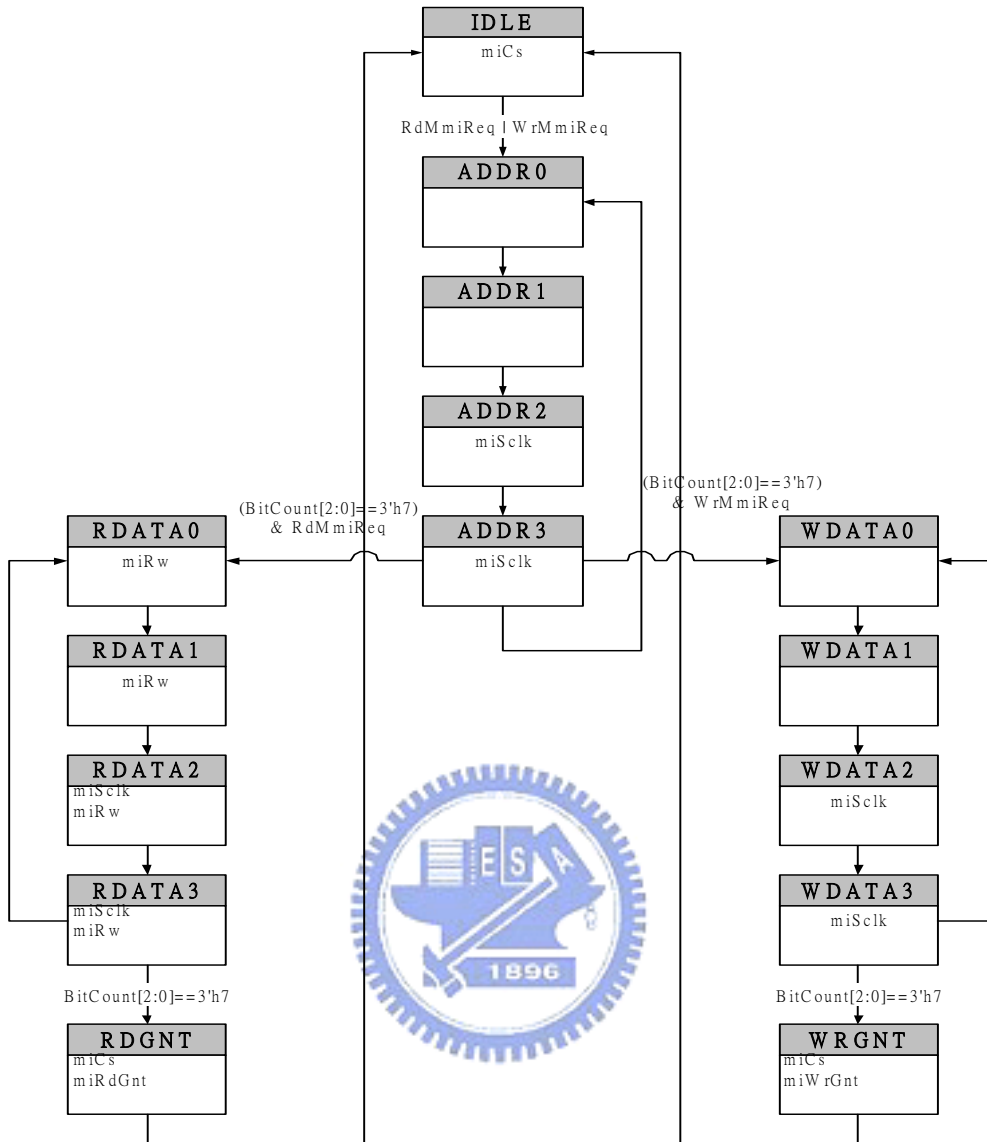


Figure 3- 34 The Flow Chart of mictrl Module FSM

3.6 RF Interface Implementation

We based on Intersil HFA3683 chip to design the RF interface control signal.

Its Pin Assignment can be seen in Figure 3-35.

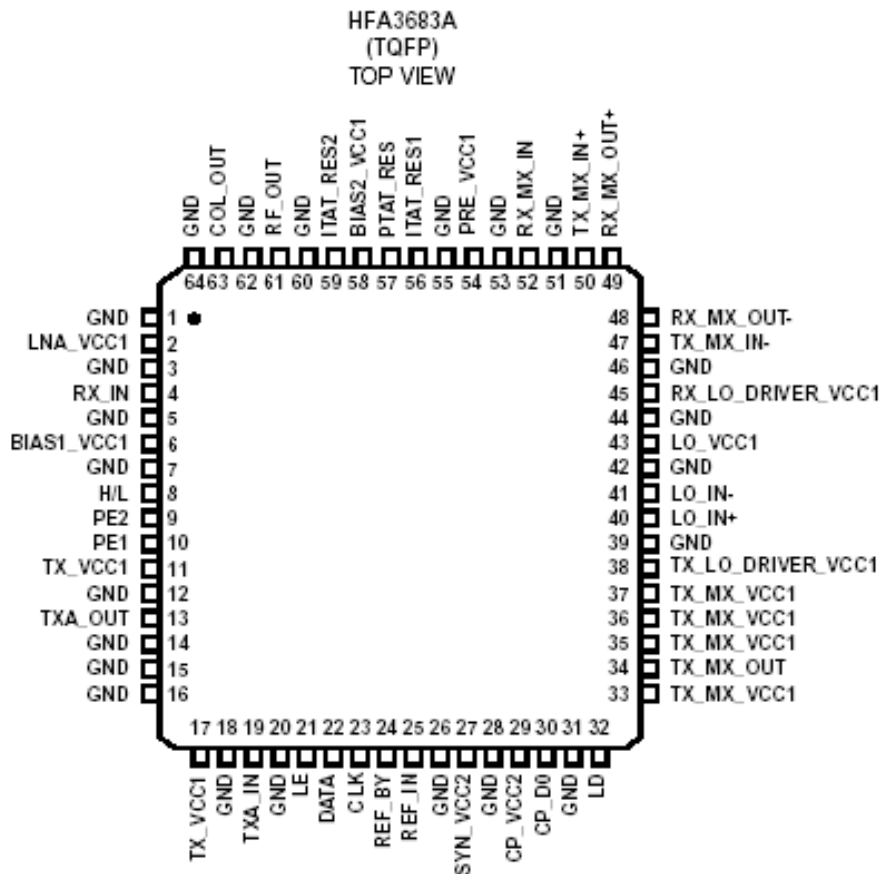
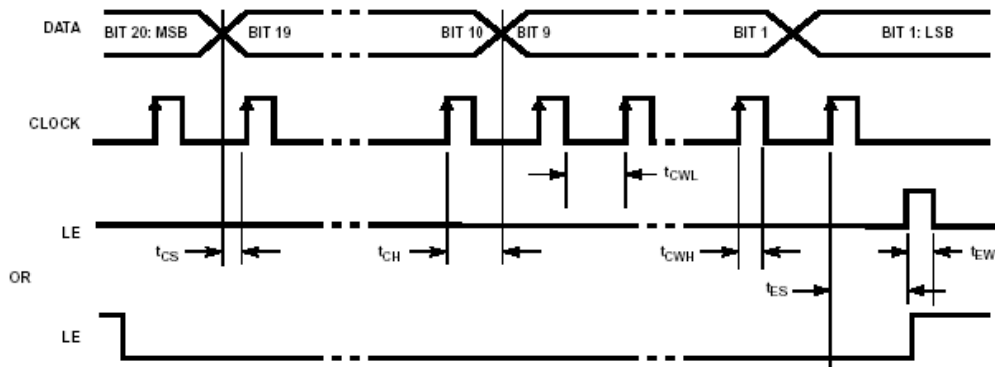


Figure 3- 35 HFA3683A Pin Assignment



WMAC can access the internal registers of HFA3683A through CLOCK, DATA and LE signals. CLOCK is the clock for DATA serial bus. The data on DATA bus is clocked at rising edge. DATA is a serial data bus, which is used to transfer address and data to the internal registers of HFA3683A chip. LE is synthesizer latch enable; the serial interface is active when LE is Low level. And the serial is latched into the defined registers on the rising edge of LE. The timing relationships of there signals are illustrated in Figure 3-36.



- NOTES:
9. Parenthesis data indicates programmable reference divider data.
 10. Data shifted into register on clock rising edge.
 11. Data is shifted in MSB first.

Figure 3- 36 HFA3683A serial data input timing

The definition of internal registers is list in table 3-3.

PLL Synthesizer Table

SERIAL BITS	REGISTER DEFINITION																			
	LSB 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	MSB
R Counter	0	0	R(0)	R(1)	R(2)	R(3)	R(4)	R(5)	R(6)	R(7)	R(8)	R(9)	R(10)	R(11)	R(12)	R(13)	R(14)	X (Don't Care)	X	X
A/B Counter	0	1	A(0)	A(1)	A(2)	A(3)	A(4)	A(5)	A(6)	B(0)	B(1)	B(2)	B(3)	B(4)	B(5)	B(6)	B(7)	B(8)	B(9)	B(10)
Operational Mode	1	0	M(0)	0	M(2)	M(3)	M(4)	M(5)	M(6)	M(7)	M(8)	0	0	0	0	M(13)	M(14)	M(15)	X	X

Reference Frequency Counter/Divider

BIT	DESCRIPTION
R(0-14)	Least significant bit R(0) to most significant bit R(14) of the divide by R counter. The Reference signal frequency is divided down by this counter and is compared with a divided LO by a phase detector.

LO Frequency Counters/Dividers

BIT	DESCRIPTION
A(0-6)	Least significant bit A(0) to most significant bit A(6) of a 7-bit Swallow counter and LSB B(0) to MSB B(10) of the 11-bit divider. The LO frequency is divided down by [P*B+A], where P is the Prescaler divider set by bit M(2). This divided signal frequency is compared by a phase detector with the divided Reference signal.
B(0-11)	

Operational Modes

BIT	DESCRIPTION				
M(0)	[PLL_FE], Phase Lock Loop Power Enable. 1 = Enable, 0 = Power Down. Serial port always on.				
M(2)	Prescaler Select. 0 = 32/33, 1 = 64/65				
M(3) M(4)	Charge Pump Current Setting	M(4)	M(3)	OUTPUT SINK/SOURCE	
		0	0	0.25mA	
		0	1	0.50mA	
		1	0	0.75mA	
		1	1	1.00mA	
M(5) M(6)	Charge Pump Sign	M(6)	M(5)		
		0	0	Source Current if LO/ [P*B+A] < Ref/R	
		0	1	Source Current if LO/ [P*B+A] > Ref/R	
M(7) M(8) M(13)	LD Pin Multiplex Operation	M(13)	M(8)	M(7)	OUTPUT AT PIN LD
		0	0	X	Lock Detect Operation
		0	1	X	Short to GND
		1	0	X	Serial Register Read Back
		1	1	0	Ref. Divided by R Waveform
		1	1	1	LO Divided by [P*B+A] Waveform
M(14) M(15)	Charge Pump Operation/Test	M(15)	M(14)	OPERATION/TEST	
		0	0	Normal Operation	
		0	1	Charge Pump Constant Current Source	
		1	0	Charge Pump Constant Current Sink	
		1	1	High Impedance State	

Table 3- 3 Register definition of HFA 3683A

Rfif.v module is used to control the operation of external RF chip. It includes two state machines, RFIFSM0 and RFIFSM1. Its Pin Assignment is shown in Figure 3-37. And flow chart is shown in Figure 3-38, 3-39.

RFIFSM0 responds for generating RFTXPE signal to active the transmit circuit of RF chip. RFIFSM1 responds for generating RFSYNCLK and RFLE signal to write the register of HFA3683A. RFSYNCLK is the clock for RFSYNDATA serial bus. The data on RFSYNDATA bus is clocked at rising edge. When RFLE is low and the serial data is latched into defined register on the rising edge of RFLE.

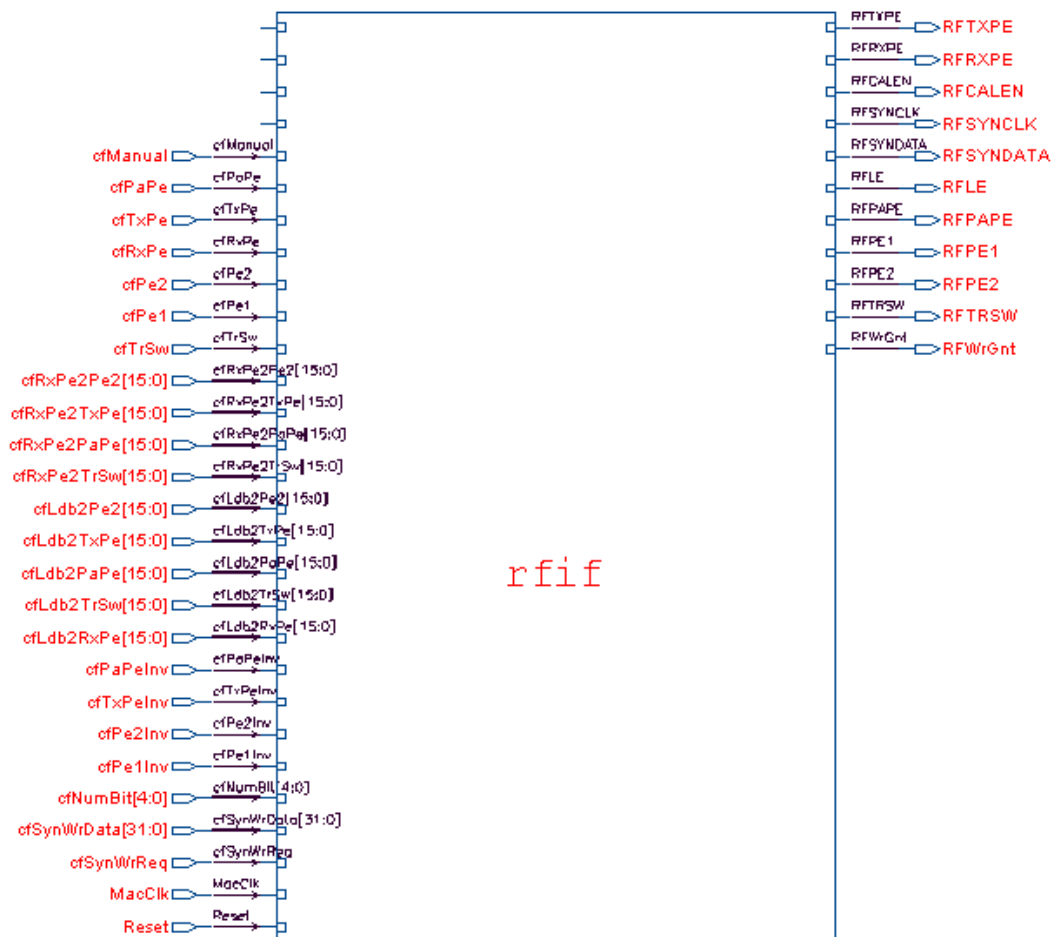


Figure 3- 37 rfif Module Pin Assignment

RXIFFSM0

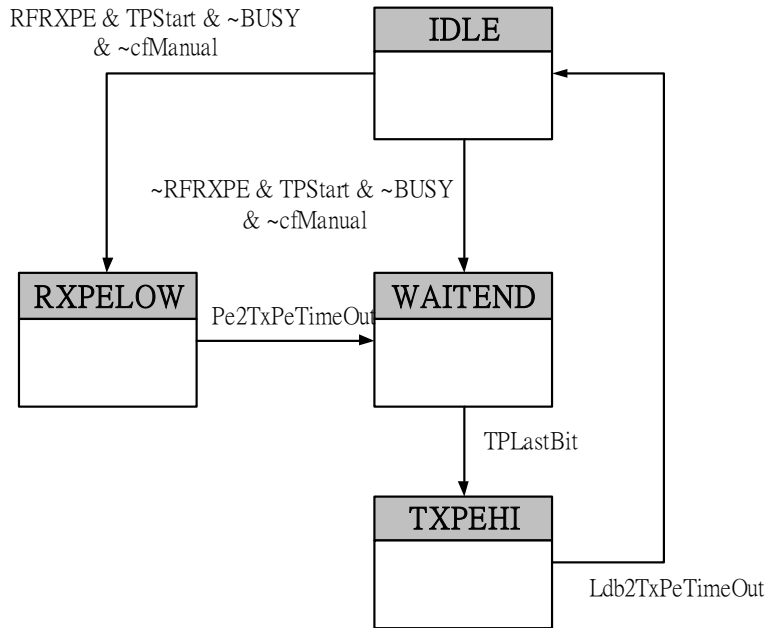


Figure 3- 38 rfif module FSM flow chart (0)

RXIFFSM1

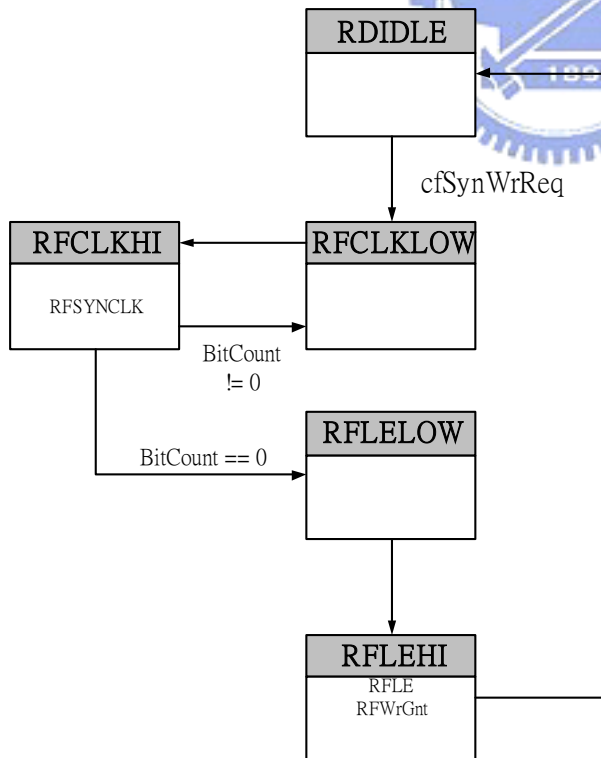


Figure 3- 39 rfif module FSM flow chart(1)

Chapter 4

Function Simulation and Verification

4.1 802.11 MAC Simulation Environment and Test Plan

For the function verification, we developed two pseudo-models to simulate the operation of physical BBP chip and RF chip. Figure 4-1 shows the Simulation environment. It consists of three components, a test bench, a wmac.v, and two simulation models; BBP.v and BBPMAC.v.

In test bench, Initial is used to setup initial value of Simulation environment. Testing task be used to control the testing pattern generator.

BBPMAC.v is a pseudo Wireless MAC model. It can accept the 802.11 protocols from WMAC.v, and response it. Or it can generate some 802.11 packets to test WMAV.v how to handle the protocol. Its Pin Assignment can be seen in Figure 4-2. It includes two major modules, the Receive module and Transmit module. Figure 4-3 show the flow of checking received frame from WMAC . It can analyze the received packets from WMAC and response with correct packets, such as CTS, ACK packets. Or we can setup BBPMAC.v to ignore some fields of received packets from WMAC. This function can test the WMAC how to deal with protocol error. Figure 4-4 show the flow of generating a frame to test WMAC function. These frames include data packets, RTS packets or beacon packets.

BBP.v is a pseudo BBP register model. It can accept the register read/write command from WMAC.v and response it with a correct value.

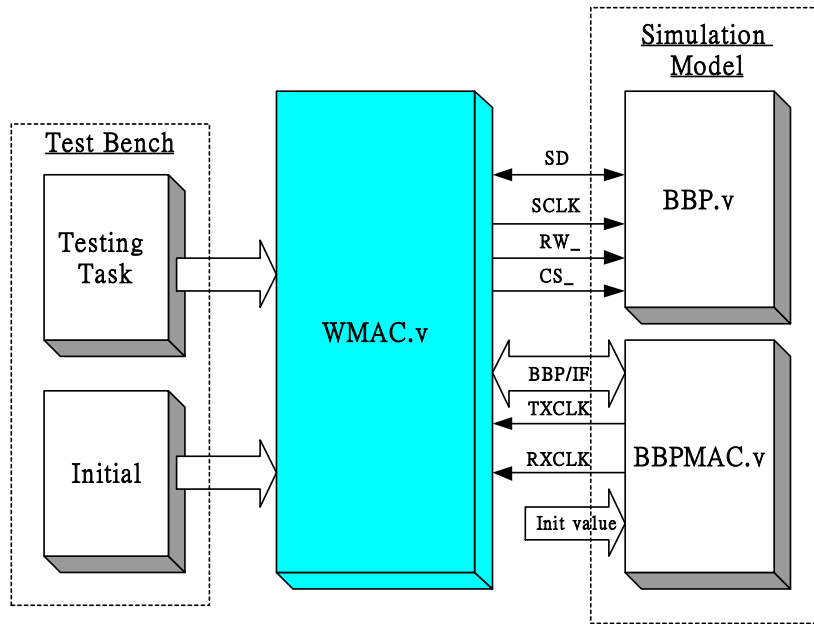


Figure 4- 1 Testing Environment

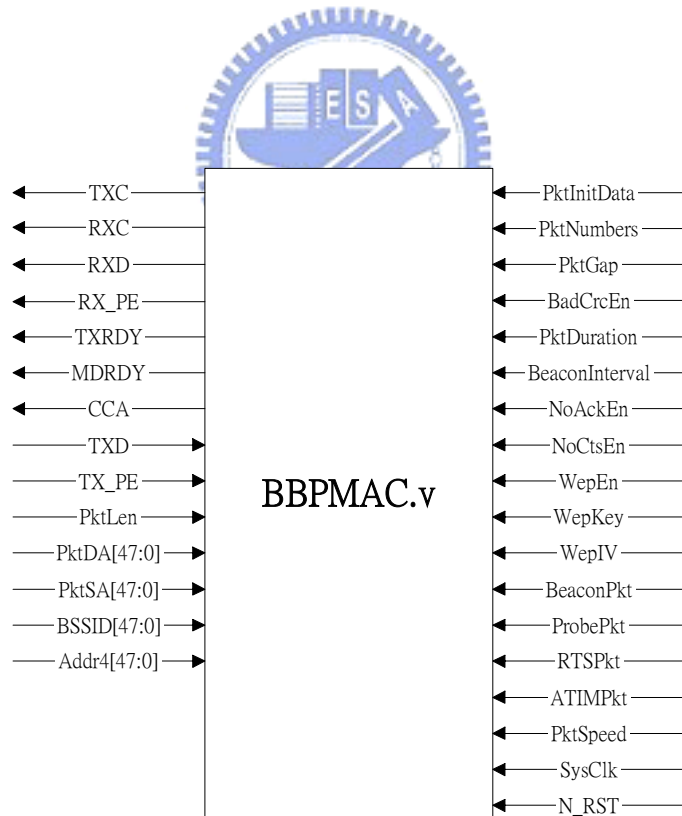


Figure 4-2 Pin Assignment of BBPMAC.v

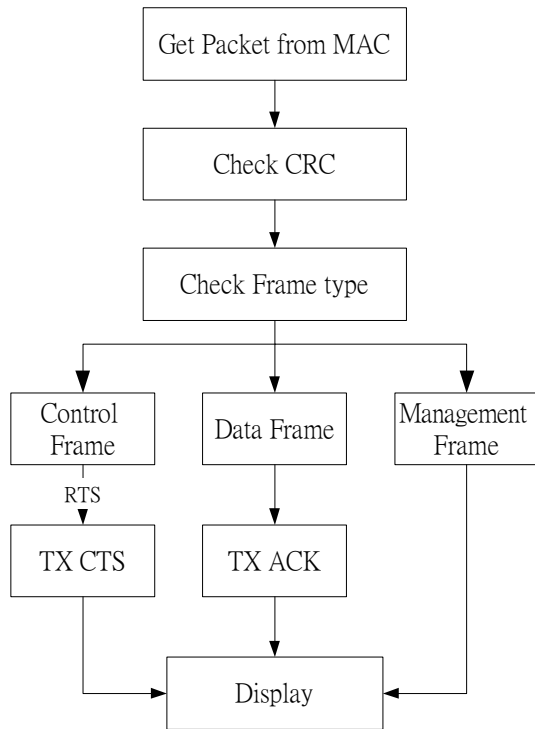


Figure 4-3 Receive function of BBPMAC.v

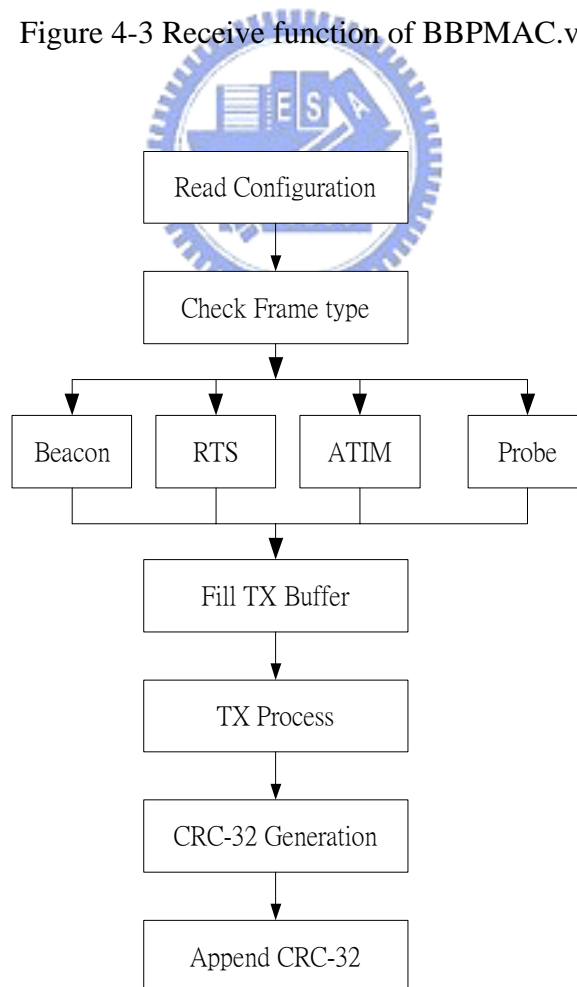


Figure 4-4 Transmit function of BBPMAC.v

The Test plan is list in table 4-1. I divide it into five testing group.

DCF Receive, which is used to verify the receive function of DCF

DCF Transmit, which is used to verify the transmission function of DCF.

BBP control group, which is used to verify the BBP control function.

RF control group, which is used to verify the RF control function.

WEP control group, which is used to verify WEP function.

Function	Case	Condition	Target	Remark
DCF Receive	Receiving a MPDU Packet	DA == Network address	Response a ACK packet	4.2.1
	Receiving a RTS Packet	DA == Network address	Response a CTS packet	4.2.2
	Receiving a RTS Packet	DA != Network address	Setup NAV	4.2.3
	Receive a beacon packet	In infrastructure station mode	Setup TBTT timer	4.2.4
DCF Transmit	Transmitting a MPDU Packet	DA = Receiver's network address	Receiving a ACK packet	4.3.1
	Transmitting a MPDU Packet (using RTS)	Using RTS Mechanism	Receiving a CTS packet and ACK packet	4.3.2
	Transmitting a MPDU Packet (Retry/Abort)	1. Retry limit = 3 2. No ACK	1. Retry three times 2. Abort this MPDU	4.3.3
	Transmitting a MPDU Packet (back off)	Channel is busy	Start back off mechanism	4.3.4

	Transmitting a MPDU Packet (Multicast address)	DA = Multicast address	Don't wait ACK packet	4.3.5
	Transmitting a Beacon Packet	DA = Multicast address	Don't wait ACK packet	4.3.6
BBP Control	BBP register read			4.4.1
	BBP register write			4.4.2
RF Control	1. RF control signal 2. Register write	Auto/manual		4.5
WEP	RC4 algorithm			4.6.1
	Encryption			4.6.2

Table 4-1 Test Plan

4.2 DCF Receive Function Verification

4.2.1 Receive a DATA (DA==Network Address) packet

The testing Case is used to verify the receive function. Its waveform is shown in Figure 4-5.

First, BBPMAC.v uses MDRDY signal to indicate to WMAC when a MPDU is send. WMAC will response an ACK Packet to BBPMAC when this packet didn't any error.

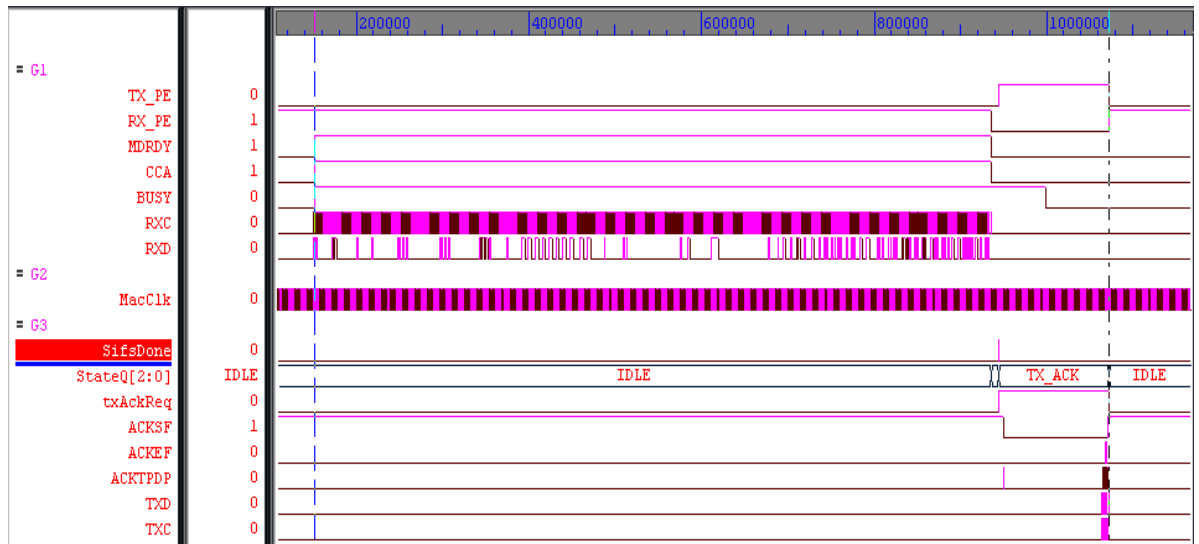


Figure 4 – 5 Receive a DATA packet

4.2.2 Receive a RTS (DA==Network Address) packet

This testing case is used to verify the function that WMAC receive a RTS packet. Its waveform is shown in Figure 4-6.

First, BBPMAC.v generate a RTS packet to WMAC. Chkpkt module will parse this packet. If its type is RTS, and its destination address is the same as local address. Then NAV timer can't be set. The WMAC replies with a CTS packet after the SIFS time.

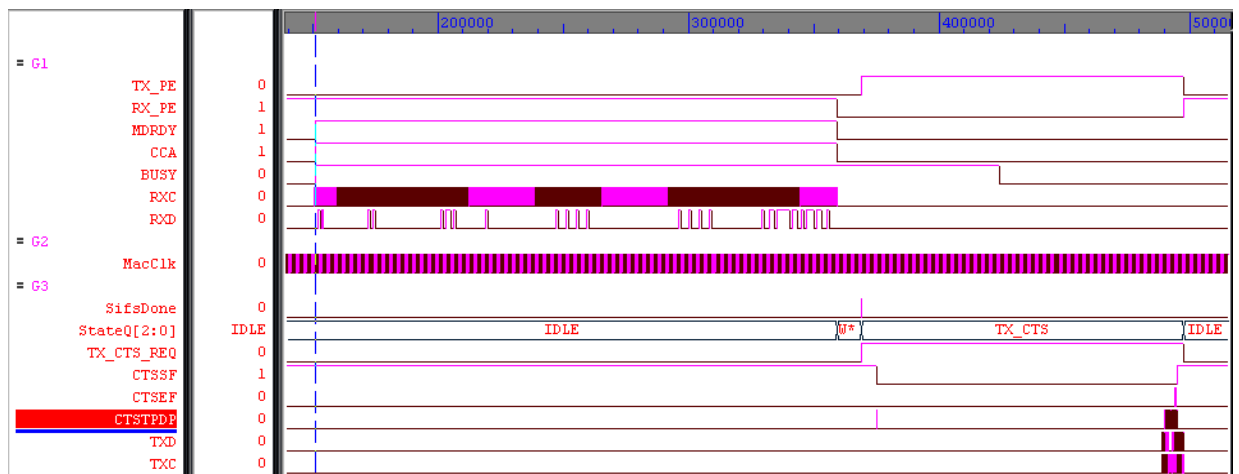


Figure 4 – 6 Receive a RTS packet (DA = Network Address)

4.2.3 NAV Timer

This testing case is used to verify the NAV timer function. Its waveform is shown in Figure 4-7.

First, BBPMAC.v generate a RTS packet to WMAC. Chkpkt module will parse this packet. If its type is RTS, and its destination address is not local address.

Then NAV timer will be set. The BUSY signal is asserted before timer comes down to zero. During this period, WMAC didn't allow send any packet to channel.

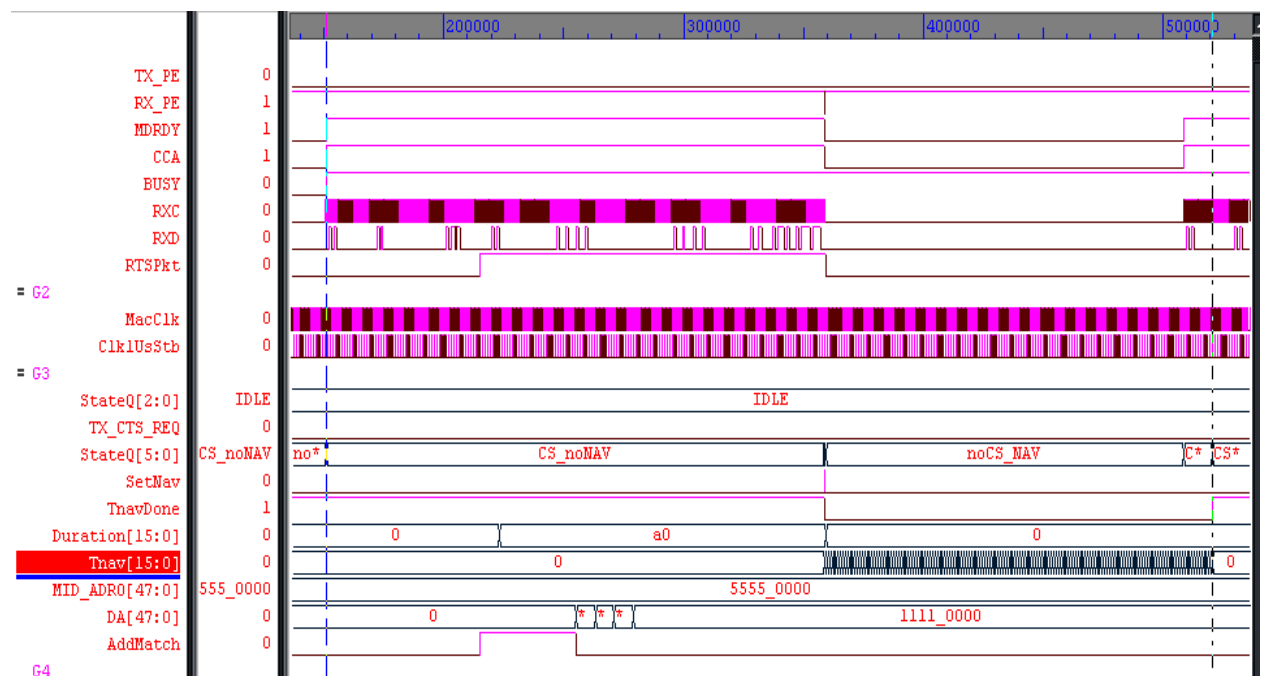


Figure 4 – 7 Receive a RTS packet (DA != Network Address)

4.2.4 Receive a Beacon packet in BSS mode.

This testing case is used to verify the function when WMAC receive a beacon packet in BSS mode. Its waveform is shown in Figure 4-8.

First, BBPMAC.v generate a beacon packet, which BSSID belong to the BSS network. Chkpkt module parses this packet. If its timestamp great than the TSF timer of local. Then update the TSF timer using timestamp value.

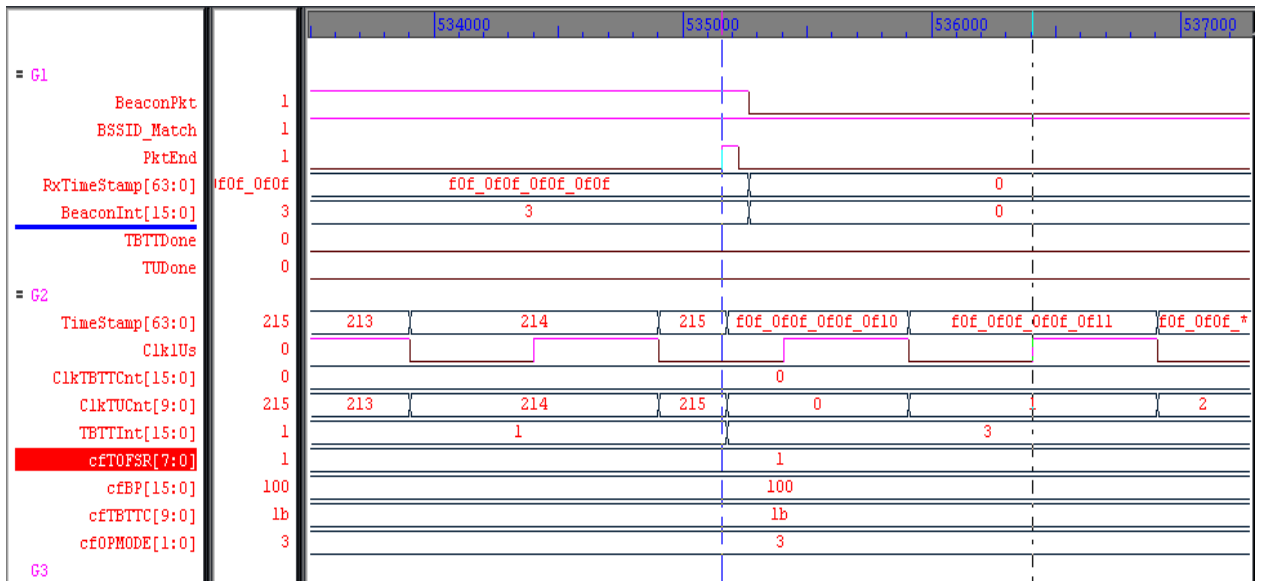


Figure 4 – 8 Receive a Beacon packet in BSS station mode

4.3 DCF Transmission Function Verification

4.3.1 Transmit a Data Packet

This testing case is used to verify the function when a MPDU want to transmit.

Its waveform is shown in Figure 4-9.

First, BM use TPSF signal to indicate to the WMAC when a MPDU want to transmit.

Then WMAC use macTPDP to load data from BM. State machine entry to IDLE state after receiving an ACK packet.

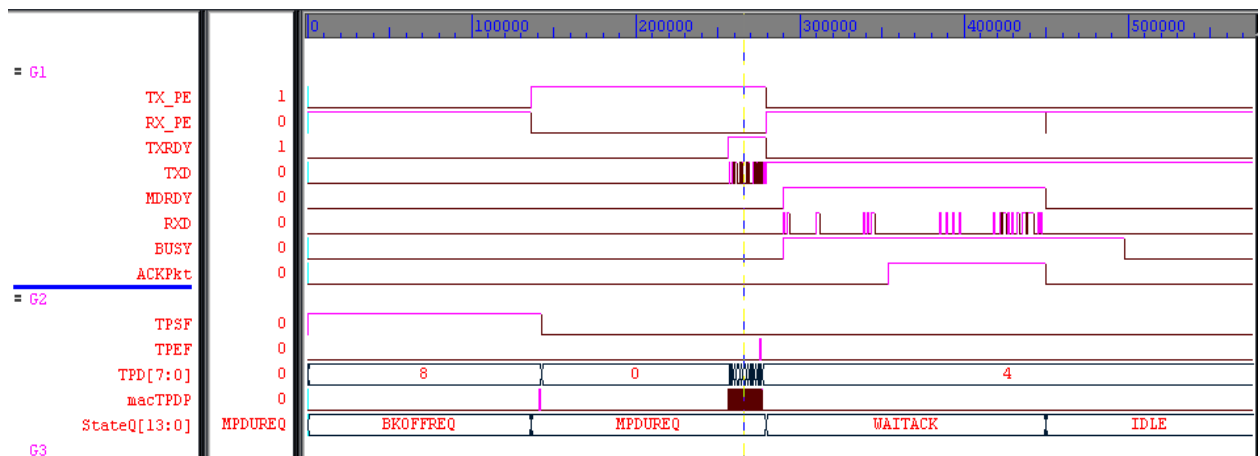


Figure 4 – 9 Transmit a DATA Packet

4.3.2 Transmit a Data Packet using RTS Mechanism.

This testing Case is used to verify the transmit function when the packet length of MPDU great than RTS threshold, MAC must be reserved the channel with RTS Packet. Then MAC must be waiting for the CTS packet that receiver response before transmit the MPDU packet. After all MPDU packets have transfer to receiver. MAC still wants to wait for the ACK packet that receiver response.

In Figure 4-10, We can observe that BM assert the TPSF to request MAC to transmit a MPDU packet, but the packet length of MPDU is great than RTS threshold, So first, State Machine entry to RTSREQ state, assert the txRtsReq to request MAC to transmit a RTS packet. When BM receives the load signal “macTPDP”, BM will cleat the TPSF to Low. Second, State Machine will entry to WAITCTS state after the RTS packet transfer completed. Third, State Machine will entry to MPDUREQ state when receiving CTS packet from receiver. Finally, MAC asserts the txMpduReq to transmit MPDU. And go back to IDLE state to wait the next transfer after receiving ACK packet from receiver.

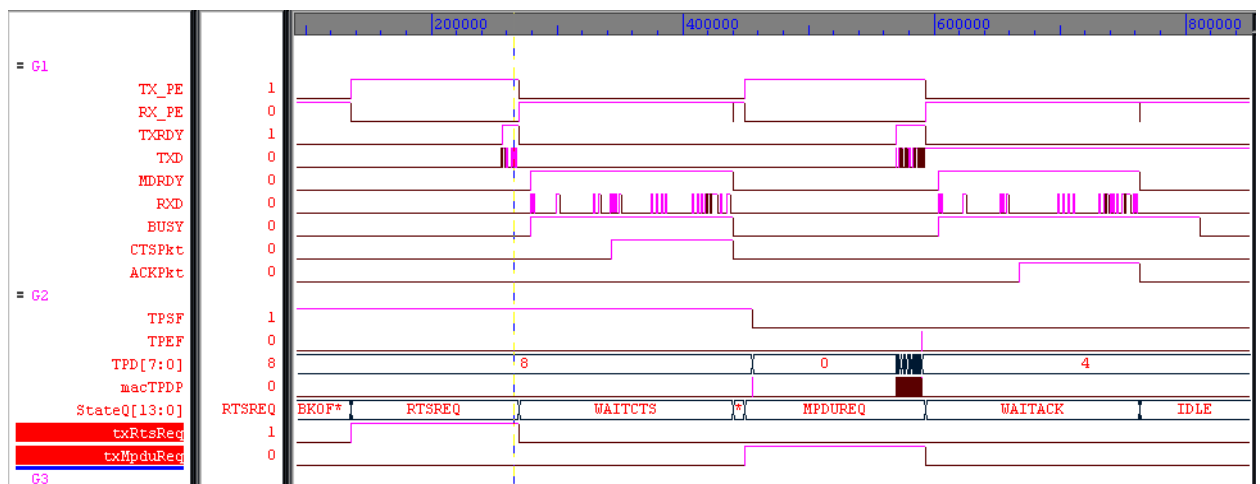


Figure 4 – 10 Transmit a Data Packet using RTS Mechanism

4.3.3 Abort the transmitted packet when lose ACK packet

This test Case is used to verify the transmit function when receiver didn't response ACK packet. Its waveform is show in Figure 4-11.

First, BM asserts the TPSF to request MAC to transmit a MPDU packet, but receiver didn't response ACK packet. So WMAC asserts TPRT to request BM to re-transmit MPDU. Second, BM repeats above process, but receiver still didn't response ACK packet. The retry count great than retry limitation, so WMAC asserts the TPAB to request BM to abort this transmission.

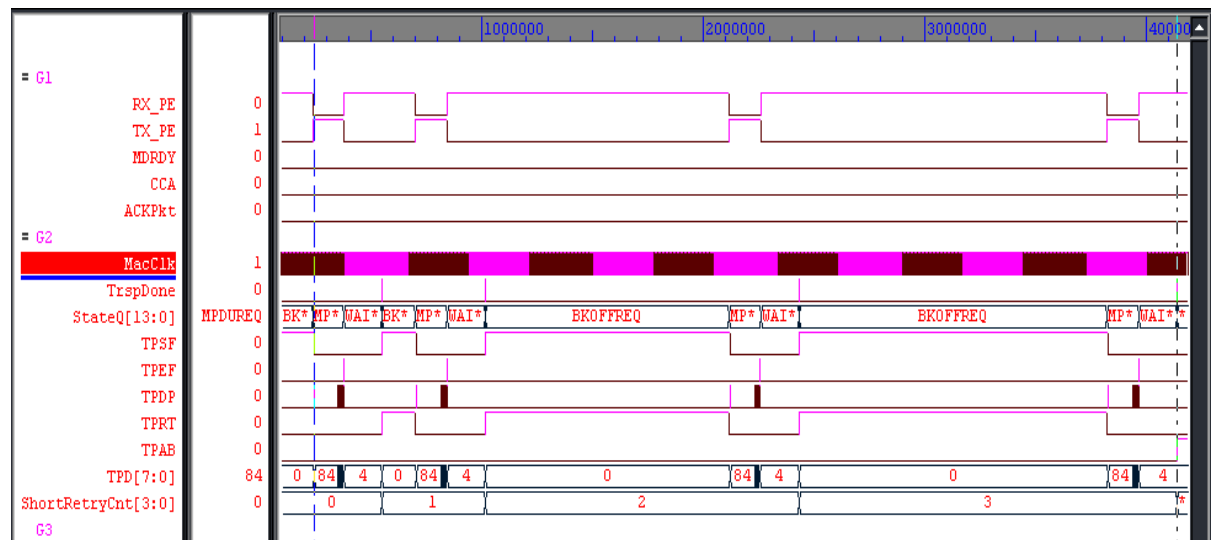


Figure 4 – 11 Abort the transmitted packet when lose ACK packet

4.3.4 Re-transmit a packet when channel is BUSY

This test case is used to verify the re-transmission function when channel state is BUSY. Its waveform is show in Figure 4-12.

First, BM asserts TPSF to request WMAC to transmit MPDU, but WMAC detect a BUSY in channel. So WMAC stop this transmission until channel change to IDLE. Then state machine will entry to BKOFFREQ state and sending this packet after waiting a random back off time.

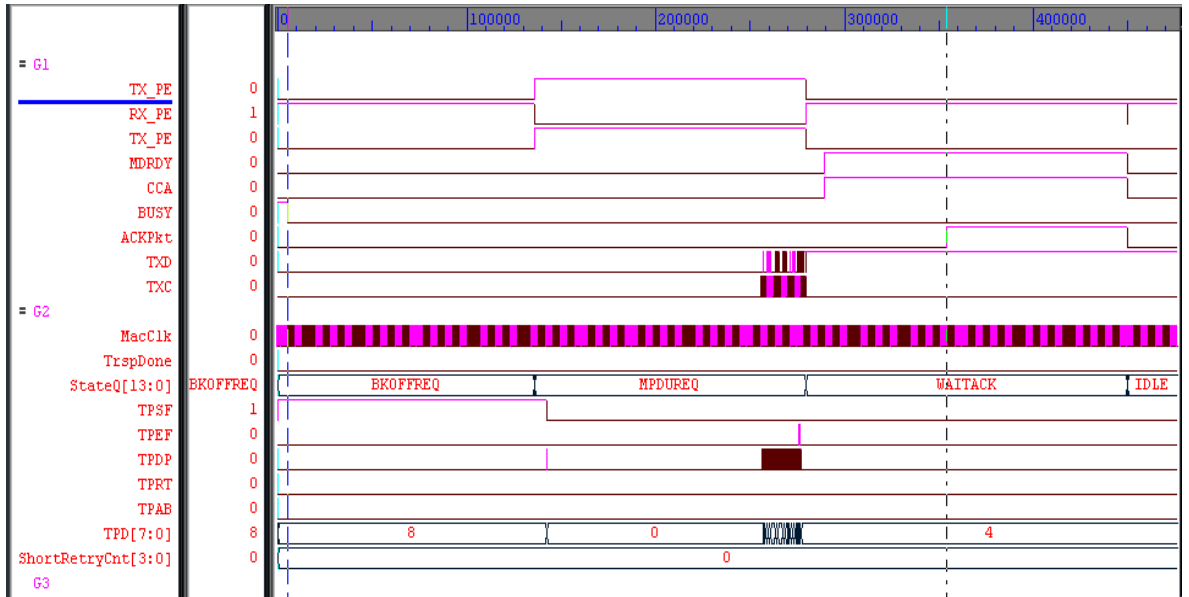


Figure 4 – 12 Re-transmit a packet when channel is BUSY

4.3.5 Transmit a Multicast packet

This test case is used to verify the transmit function when destination address of MPDU is multicast. Its waveform is shown in Figure 4-13.

First, BM asserts TPSF to request WMAC to transmit a MPDU, but the destination address of MPDU is multicast, so the state machine will enter into IDLE state after transmission. Not waiting for the response ACK packet.

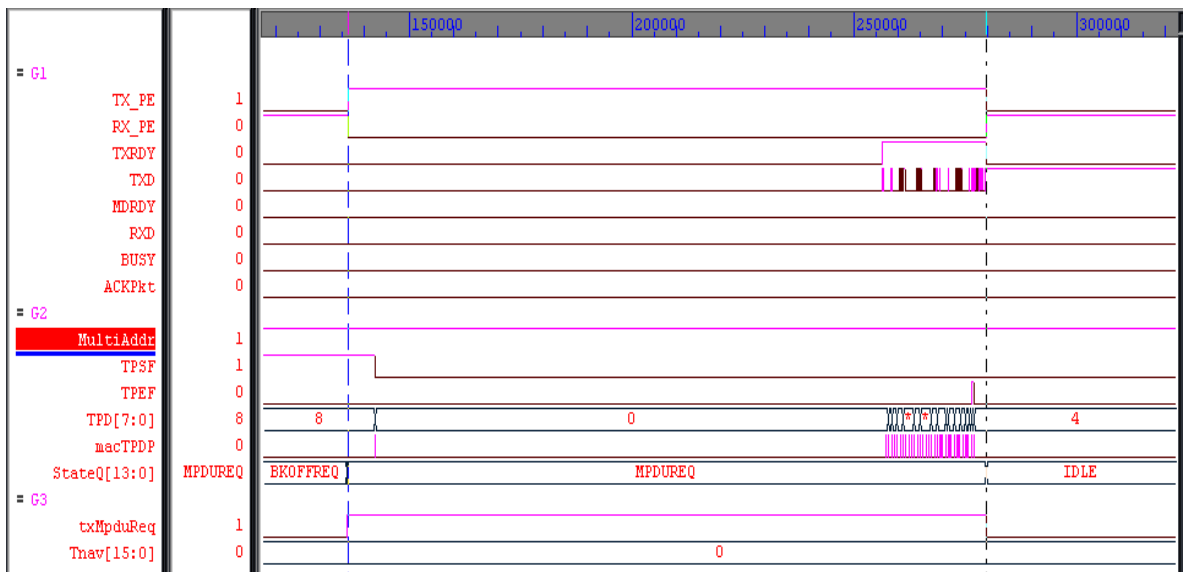


Figure 4 – 13 Transmit a Multicast packet

4.3.6 Transmit a Beacon Packet

This test case is used to verify the transmission of beacon packet. Its waveform is shown in Figure 4-14.

Once “TBTTDone” signal is asserted. The state machine will entry into BEACONREQ state and generate a “txBeaconReq” to tx_pump module.

Tx_pump module will load data from SSRAM with BEACONTPDP signal.

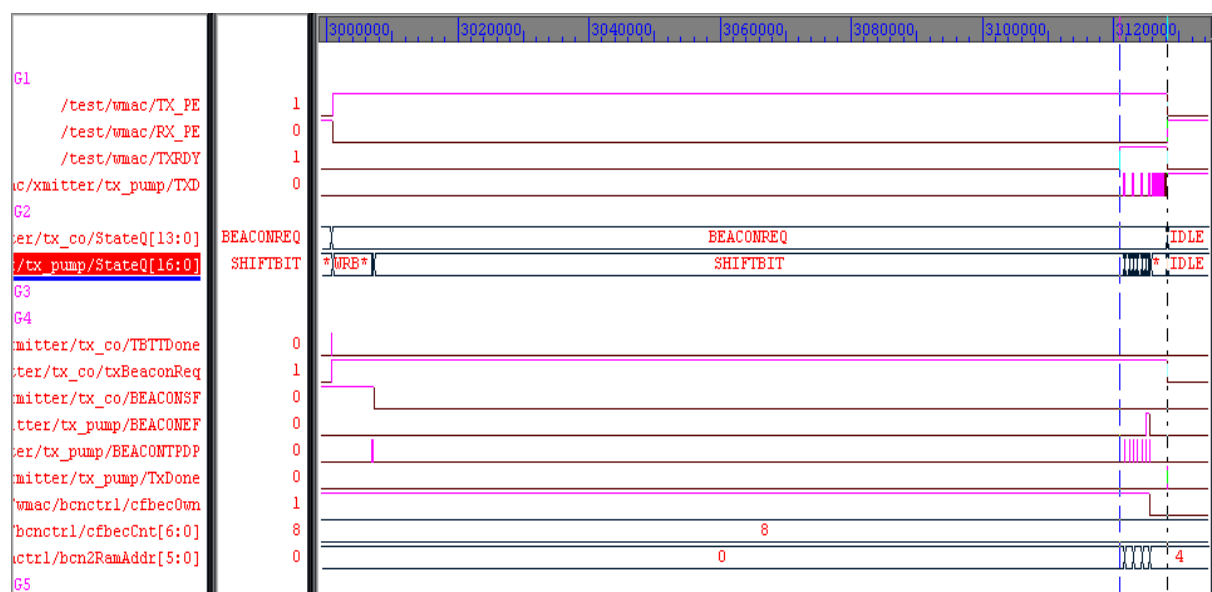


Figure 4 – 14 Transmit a Beacon Packet

4.4 BBP Interface Function Verification

4.4.1 Write BBP Register

This test case is used to verify the function that WMAC writes BBP’s register. Its waveform is shown in Figure 4-15.

Rwbbp.v will set the content of LENGHTH, RATE, SERVICE registers when WMAC require BBP to transmit a MPDU.

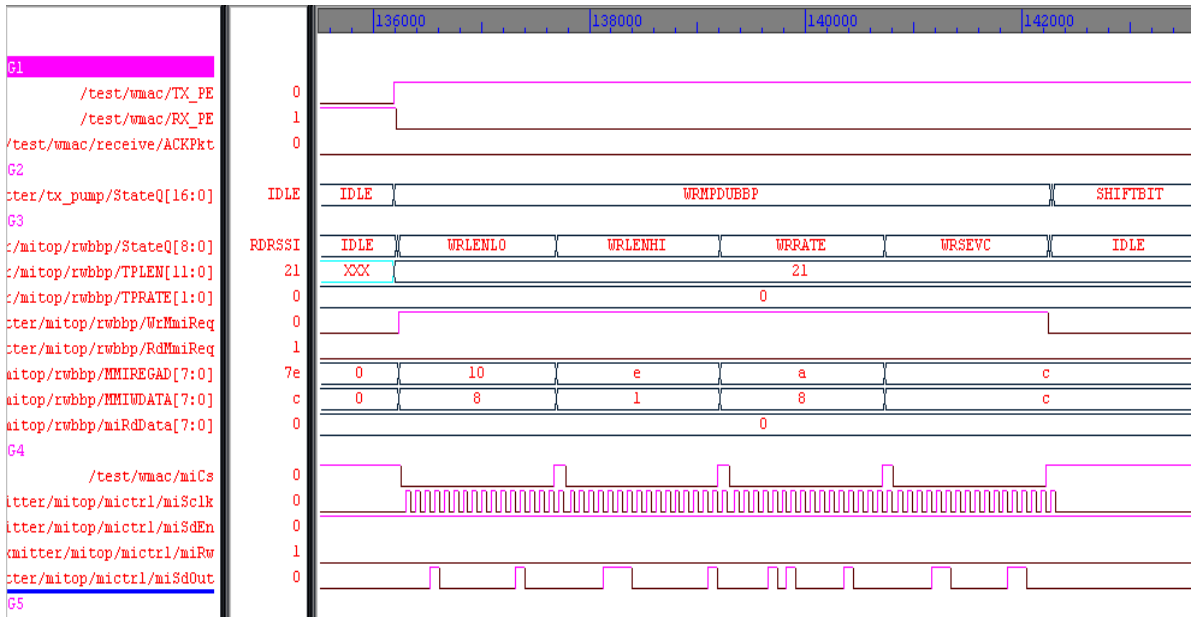


Figure 4 – 15 BBP register write timing

4.4.2 Read BBP Register (RSSI Reg)

This test case is used to verify the function that WMAC reads BBP's register. Its waveform is shown in Figure 4-16.

Mictrl.v module will read the content of RSSI register when WMAC receive a MPDU from BBP.

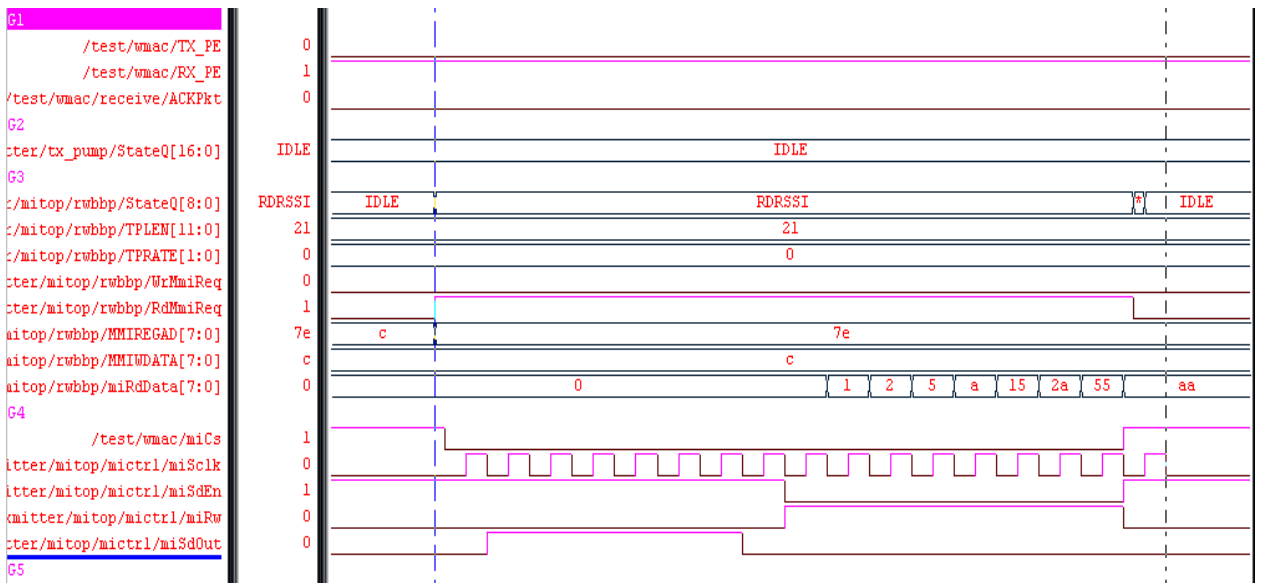


Figure 4 – 16 BBP register read timing

4.5 RF Interface Function Verification

This test case is used to verify the function of RF chip control. Its waveform is shown in Figure 4-17.

At time 44000. WMAC automatically generate the handshake signals between WMAC and RF chip, such as RFTXPE, RFPAPE, RFPE2 and RFTRS.

Time 44000 to 60000. WMAC manually generate the handshake signal between WMAC and RF chip.

From time 80000 to 110000. WMAC writes that registers in RF chip using RFSYNCLK and RFSYNDATA signal.

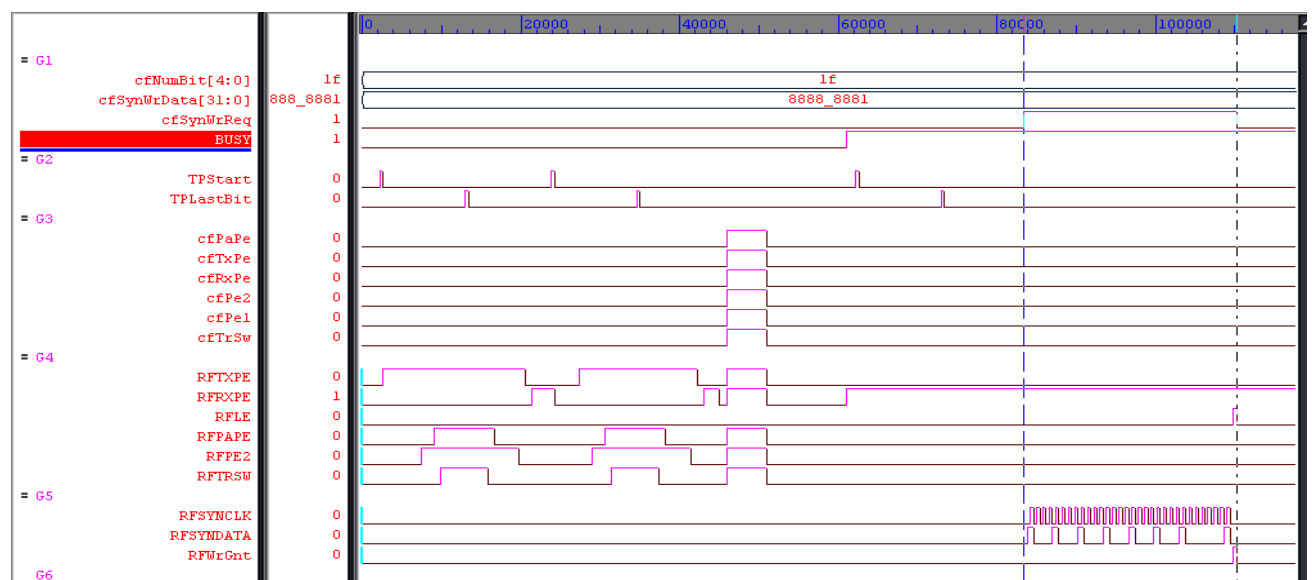


Figure 4 – 17 RF chip control timing

4.6 WEP Function Verification

4.6.1 RC4 Algorithm Verification

This test case is used to verify the RC4 algorithm. Its waveform is shown in Figure 4-18.

In initialization stage, RC4.v will fills that values in S-BOX RAM use the number of 0 ~ N-1. In scrambling stage. First, RC4.v will read two values from

S-BOX RAM, then exchange it. Finally, write its value into S-BOX RAM.

The content of S-BOX RAM after performing round one is shown in Table 4-2.

The content of S-BOX RAM after performing round two is shown in Table 4-3

Compare with table 4-2, 4-3 and Figure 4-15. We can prove the correctness of RC4 engine.

Round 1:

$i_1 = 0$
$j_1 = j_0 + S_0[0] + k[0] = 0 + 0 + 23 = 23$
Swap ($S_0[0], S_0[23]$)
$S_1 = \{23, 1, 2, 3, \dots, 22, 0, 24, \dots, 254, 255\}$

Table4-2 WEP Round 1

Round 2:

$i_2 = 1$
$j_2 = j_1 + S_1[1] + k[1] = 23 + 1 + 89 = 113$
Swap ($S_1[0], S_1[23]$)
$S_2 = \{23, 113, 2, 3, \dots, 22, 0, 24, \dots, 112, 1, 114, \dots, 254, 255\}$

Table4-3 WEP Round 2

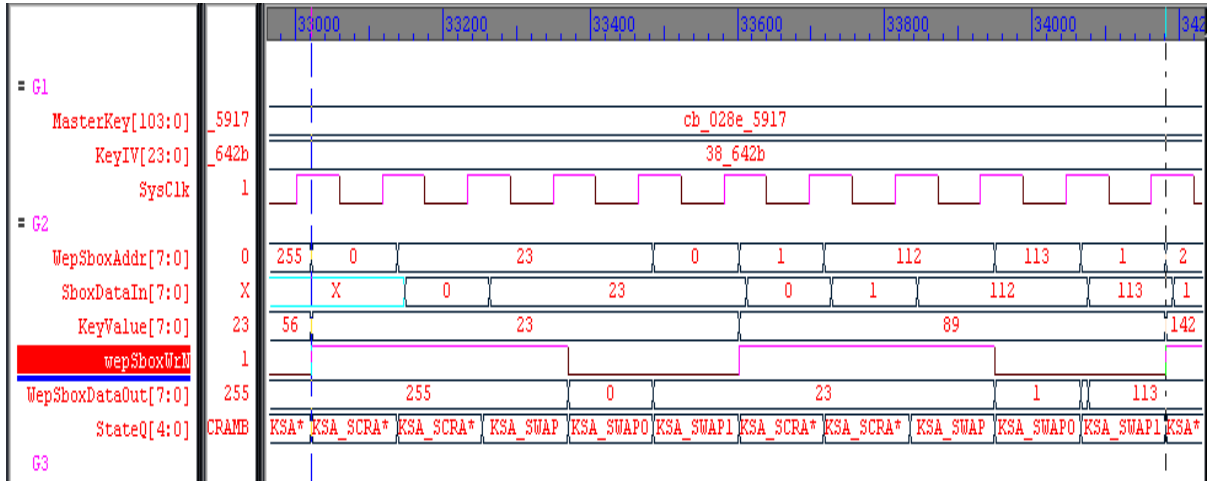


Figure 4 – 18 RC4 KSA

4.6.2 WEP Verification

This test case is used to verify the WEP encryption. Its waveform is shown in Figure 4-19.

First, BM use `bmTPSF` to indicate to the MAC when a MPDU want to transmit, then WMAC use `wepTPDP` to indicate to the BM when WMAC can load data. Before data payload 1,2,3, there data belong to header. So WEP didn't encrypt those data when `ENCRYPPhase` is Low. During data payload 1,2,3. Those data belong to payload, so WEP encrypt those data when `ENCRYPPhase` is High.

BM use `bmTPEF` to indicate to WEP module when a MPDU is transmitting completion. WEP module will appends its ICV field to the transmitted frame.

Chapter 5

FPGA Synthesis and P&R

5.1 FPGA Synthesis and P&R

We synthesizing the RTL code in Xilinx FPGA (v2000efg1156-8) with Foundation V3.1 FPGA Express, and give clock rate and operation condition constrains. After that, we get the gate level netlist. It stores to an EDIF file. In FPGA P&R stage, we use Foundation V3.1 Design Manager to automatically place and route with gate level netlist and perform the STA for timing analysis. The flow is show in figure 5-1.

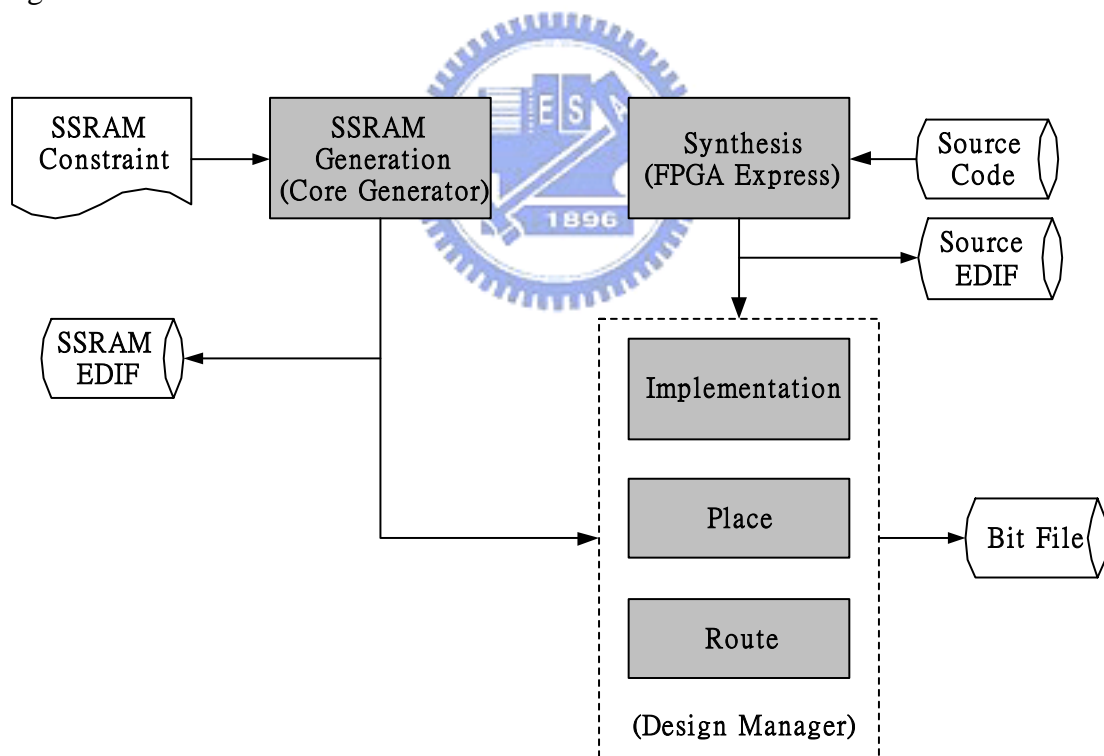


Figure 5-1 FPGA Place & Route Flow Chart

5.2 FPGA Synthesis and Timing Report

The detail chip layout and gate count are list in Table 5-1. The total equivalent gate count is 51,029 gates. The detail timing report is list in table 5-2. The critical path of whole chip is 20.635ns. So this design can work at clock speed = 48.461MHz at v2000efg1156-8 FPGA device.

Synthesis Tool: Foundation V3.1 FPGA Express.

Place & Route Tool: Foundation V3.1 Design Manager.

Design Summary:			
Number of Slices:	1,897 out of 19,200		9%
Number of Slices containing			
Unrelated logic:	0 out of 1,897		0%
Total Number Slice Registers:	1,406 out of 38,400		3%
Number used as Flip Flops:	1,357		
Number used as Latches:	49		
Total Number 4 input LUTs:	3,176 out of 38,400		8%
Number used as LUTs:	3,162		
Number used as a route-thru:	14		
Number of bonded IOBs:	558 out of 804		69%
Number of Block RAMs:	1 out of 160		1%
Number of GCLKs:	4 out of 4		100%
Number of GCLKIOBs:	2 out of 4		50%
Total equivalent gate count for design:	51,029		
Additional JTAG gate count for IOBs:	26,880		
Removed Logic Summary:			

8 block(s) removed

271 block(s) optimized away

8 signal(s) removed

Table 5-1 v2000efg1156-8 Synthesis Report

Constraint	Requested	Actual	Logic Levels
TS_xmitter_mitop_rwbbp_N2456 = PERIOD TIM EGRP "xmitter_mitop_rwbbp_N2456" 16 nS HIGH 8 nS	16.000ns	10.896ns	5
TS_MacClk = PERIOD TIMEGRP "MacClk" 16nS HIGH 8 nS	16.000ns	19.355ns	13
TS_SysClk = PERIOD TIMEGRP "SysClk" 16nS HIGH 8 nS	16.000ns	9.751ns	5
OFFSET = IN 16 nS BEFORE COMP "MacClk"	16.000ns	20.088ns	13
OFFSET = OUT 16 nS AFTER COMP "SysClk"	16.000ns	14.408ns	2
OFFSET = OUT 16 nS AFTER COMP "MacClk"	16.000ns	15.105ns	2
Maximum path delay from/to any node:	12.197ns		
Minimum input arrival time before clock:	20.088ns		
Minimum output required time after clock:	15.105ns		
Minimum period:	20.635ns (Maximum frequency: 48.461MHz)		

Table 5-2 v2000efg1156-8 Timing Report

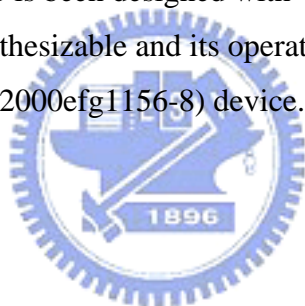
Chapter 6

Conclusion

In this thesis, we implement the MAC and WEP function of the IEEE 802.11 in hardware. The hardware-based MAC design and implement using Specification and Description Language (SDL) of 802.11 standard. WEP engine uses FIFO RAM and PLCP preamble time to recover the overhead of processing key-scheduling algorithm (KSA) at RC4 algorithm. In particular, we show that the use of 256 Byte FIFO RAM is enough. The performance of this WEP engine can up to the 54Mbps channel-speed.

The pseudo 802.11 MAC is used to test the functionalities of the MAC design. After through testing, our design is proven to meet the functional requirements of a BSS network specified in the IEEE 802.11 MAC layer specification.

The circuit level of MAC is been designed with Verilog RTL and implemented in Xilinx FPGA. It is fully synthesizable and its operation frequency can scale up to 44MHz at the Xilinx FPGA (v2000efg1156-8) device. The total equivalent gate count is 51,029.



Reference

- [1] “IEEE 802.11g standard 2003 Edition” 2003
- [2] “Wireless LAN medium access control (MAC) and physical layer (PHY) specification. IEEE Standard 802.11, 1999 Edition” 1999
- [3] Marvell Semiconductor Inc, “Libertas Wireless LAN 802.11g (b) access point, Wireless Getaway chipset 88W8000G and 88W8510”.
- [4] Agere System Inc, “WaveLan 802.11 a/b/g chip set.
- [5] MAC implementation for IEEE 802.11 wireless LAN
Youjin Kim; Haewon Jung; Hyeong Ho Lee; Kyong Rok Cho;
ATM (ICATM 2001) and High Speed Intelligent Internet Symposium, 2001.
Joint 4th IEEE International Conference on , 22-25 April 2001
Pages: 191 – 195
- [6] 802.11a MAC layer: firmware/hardware Co-design
Yeong, J.H.; Rao, X.M.; Shajan, M.R.; Wang, Q.; Lin, C.Y.; Qu, J.X.H.;
Information, Communications and Signal Processing, 2003 and the Fourth
Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint
Conference of the Fourth International Conference on, Volume: 3 , 15-18 Dec.
2003 Pages: 1923 – 1928
- [7] “Design and Implementation of IEEE 802.11 MAC Controller : Transmitter Part” Fu-Shung Lin. Institute of Communication Engineering College of Electrical and Computer Science of National Chiao Tung University. Master thesis
- [8] Intersil Semiconductor Inc, “Direct Sequence Spread Spectrum Baseband Processor HFA3861B”.
- [9] Intersil Semiconductor Inc, “2.4G RF/IF Converter and Synthesizer HFA3683”.

[10] Xilinx FPGA Vertex Data Sheet (www.xilinx.com)

[11] R.L.Rivest, "The RC4 Encryption Algorithm," RSA Data Security Inc., March 12, 1992



Appendix A The boundary signals of chip

□ Output Signals

Destination	Name
Group A To RF Chip	RFTXPE, //IO RFRXPE, //IO RFCALEN, //IO RFSYNCLK, //IO RFSYNDATA, //IO RFLE, //IO RFPAPE, //IO //RFRAPE, //IO RFPE1, //IO RFPE2, //IO RFTRSW, //IO
Group B To Baseband Processor (BBP)	RX_PE, TX_PE, TXD, miSclk, miRw, miCs, miSdOut, miSdEn,
Group C To Register Sets	RFWrGnt, // to CFG mmiRdGnt, mmiWrGnt, miRdData, miRSSI, macTSFT, //CFG, TSFT Timer[63:0] ? macLSI, //CFG, Listen Interval ShortRetryCnt, LongRetryCnt, bcnClrOwn, bcRamData,

Group D To Buffer Manager (BM)	macRxSa,	//RX
	macWepIv,	//RX
	macKeyID,	//RX
	macSaStb,	//RX
	macRxWep,	//RX
	macFrameStr,	//RX
	macFrameEnd,	//RX
	macRPD,	//RX
	macRPDV,	//RX
	macRPSF,	//RX
	macRPEF,	//RX
	macRSVP,	//RX
	macRPGOOD,	//RX
	macRSV,	//RX
	macTPDN,	//TX
	macTPDP,	//TX
	macTPRT,	//TX
macTPAB,	//TX	
macTSVP,	//TX	
macTSV	//TX	

□ **Input Signal**

Source	Name
Group E From Buffer Manager (BM)	TPD, //TX
	TPSF, //TX
	TPEF, //TX
	TPUR, //TX
	TPUD, //TX
	TPLEN, //TX, transmitted packet length
	TPRATE, //TX, transmit rate
	TPProbeRsp,
	MultiAddr,
	MoreFrag,
Group F From Baseband Processor (BBP)	TXC, //IO
	TXRDY, //IO
	CCA, //IO
	RXD, //IO

	RXC, //IO MDRDY, //IO IoSdIn,
Group G From Register Sets	cfMaxPktLen, cfPassBad, cfDIFS, //CFG, DIFS Timer cfEIFS, //CFG, EIFS Timer cfMacAddr0, //[47:0] cfMacAddr1, cfMacAddr2, cfMacAddr3, cfHashTab, //[63:0] cfBSSID, cfPROM, cfAMCP, cfFBCP, cfXMTEN, //CFG, Transmit Enable cfRCVEN, //CFG, Receiver Enable cfPlcpEn, //CFG, PLCP Header Enable cfBEACONEN, //CFG, Auto generate Beacon Enable cfTOFSR, //CFG, RX TSFT offset cfTOFST, //CFG, TX TSFT offset cfMTMLT, //CFG, Max Tx MSDU life time cfBP, //CFG, Beacon Period cfTBTT, //CFG, TBTT compensation cfWrMmiReq, cfRdMmiReq, cfMMIREGAD, //MMI Register Address cfMMIWDATA, cfPreamble, //0 : long preamble, 1: short preamble cfLBM, //CFG, Loop back mode cfPbcc, //CFG, PBCC mode cfBSCRATE, //CFG, Basic Tx Rate cfNeedRts, cfCtsTimeOut, cfAckTimeOut, cfRtyLimit, //CFG, Retry Limit cfOPMODE, //CFG, Operation Mode

cfCWMax,	//Contention windows Max (1023)
cfCWMin,	//Contention windows Min (7)
cfManual,	
cfPaPe,	
cfTxPe,	
cfRxPe,	
cfPe2,	
cfPe1,	
cfTrSw,	
cfRxPe2Pe2,	
cfRxPe2TxPe,	
cfRxPe2PaPe,	
cfRxPe2TrSw,	
cfLdb2Pe2,	//Last data bit to PE2
cfLdb2TxPe,	//Last Data bit to TXPE
cfLdb2PaPe,	//Last Data bit to PAPE
cfLdb2TrSw,	//Last Data bit to TRSW
cfLdb2RxPe,	//Last Data bit to RXPE
cfPaPeInv,	
cfTxPeInv,	
cfPe2Inv,	
cfPe1Inv,	
cfNumBit,	
cfSynWrData,	
cfSynWrReq,	
cfbecOwn,	//to beacon ctrl
cfbecCnt,	//to beacon ctrl
cfTableWr,	//to beacon ctrl
cfTableRd,	//to beacon ctrl
cfTableAddr,	//to beacon ctrl
cfTabAddrWrN,	//to beacon ctrl
cfTabDataWrN,	//to beacon ctrl
cfTabDataRdN,	//to beacon ctrl
cfBcnData,	
RTSSADDR,	
RTSDADDR,	
RTSDuration,	