Short Paper

A COMBINATION LOGIC DESIGN FOR A HIGH SPEED IEEE 802.11 MAC CONTROLLER

Ren-Zong Li*, Shang-Pin Huang, Fu-Shung Lin, Ming-Tsung Hong, and Po-Ning Chen

ABSTRACT

In this paper, we propose a pure combinational logic design for the implementation of IEEE802.11 Medium Access Control (MAC) protocol, in contrast to firmware implementation based on an embedded micro-engine. In order to have a timely response, a Control Frame Handler is also included in our MAC controller. A further improvement for timely manipulation on the time-critical management frames (such as Beacon, ATIM and Probe Response) is subsequently developed in our revised edition. Equipped with a self-developed PCMCIA unit, the functions of our MAC controller have been verified using two Altera EPF 10K-100 ARC240-2 FPGAs in an on-line MAC-to-MAC data exchange fashion. Experimental results show that a pure combinational logic design can easily achieve a baseband-interfacing throughput of over-100Mbps with a cost-effect gate count of 21702.

Key Words: IEEE 802.11, medium access control, wireless local area network.

I. INTRODUCTION

The current implementations of IEEE 802.11 Medium Access Control (MAC) mostly incorporate a CPU core in their integrated circuit designs, where the MAC protocol is realized through firmware implementation. Two renowned examples are the AMD79C30 and HFA3842 MAC controllers. The former employs an embedded 80188 core, while the latter incorporates a micro-programmed MAC engine. As the IEEE 802.11 MAC standard (1999) converges to DFW CSMA/CA, and no further revision on the underlying MAC standard is in process, the need for flexibility and customization of the MAC design has gradually shifted to the demand for a cost-effective design, i.e., a design that can achieve high speed at a fairly low cost. This motivated us to develop a pure combinational-logic-based MAC controller.

Different from the Ethernet standard, the MAC specified in IEEE 802.11 requires more management efforts (in addition to the basic CSMA/CA mechanism) due to the unreliable nature of wireless transmissions. The ability to handle the control frames and the management frames are therefore essential to an IEEE 802.11 MAC controller. Perhaps, this is the key reason why the firmware-based implementation approach is more prevalent on the market. For example, a timely layer-2 acknowledgement and re-transmission due to previous transmission failure are specified in the standard. To fulfill the management requirement, a separate *Control Frame Handler* circuit is designed to manipulate the timely transmission and response of the control frames, such as RTS, CTS and ACK frames. This unit closely co-works with other units under the finite-state machine on which our design is based, and complements the management functionality of IEEE 802.11. A second revision of our MAC verilog code additionally includes

*Corresponding author. (Tel: 886-3-5712121 ext. 52947; Fax: 886-3-5731663; Email: brandon@eic.nctu.edu.tw)

R. Z. Li and S. P. Huang are with the Trans. Wireless Technology Laboratories, National Chiao Tung University, Hsin Chu, Taiwan 300, R.O.C.

F. S. Lin is with the Macronix International Co., Ltd., Taiwan 300, R.O.C.

M. T. Hong is with the VXIS Technology Corporation, Taiwan 300, R.O.C.

P. N. Chen is with the Department of Communications Engineering, National Chiao Tung University, Hsin Chu, Taiwan 300, R.O.C.

the manipulation of time-critical management frames (which was previously designed to be handled by the host driver) to further enhance the system performance.

Our experimental design was carried out in two stages, which yielded two versions of MAC verilog codes. In the first stage, we only attempted to substantiate the idea of realizing the IEEE 802.11 MAC protocol using pure combinational logic, and only targeted the 1/2 Mbps basic processing speed of IEEE 802.11 MAC (Hong 1999; Lin, 1999). After its theoretical effectiveness was shown, a major revision of the previous version subsequently proceeded, which resulted in an over-100 Mbps data rate to-and-from the baseband processor (Huang, 2000). Although the current standard only specifies up to 54 Mbps nominal data rate (IEEE std. 802.11a, 1999), our experimental implementation does confirm the feasibility and cost-effectiveness of a combinational logic design of an IEEE 802.11 MAC.

It is worth mentioning that a pure combinational logic design of IEEE 802.11 MAC also facilitates the chip-level integration with the baseband processor, as we have done in our joint project with the baseband team. Since both the MAC and the baseband circuits are implemented directly using the verilog language, a cell-level joint-simulation can be readily performed in an on-line transceiving fashion, which largely ensures the workability of an integrated MAC/baseband processor.

II. BASIC DESIGN OF THE MAC CONTROLLER

1. General Description on MAC Controller Design

In the first trial, only the compulsory distribution coordination function (DCF) scheme over ad hoc topology was implemented. The supports to the optional point coordination function (PCF) scheme as well as the infrastructure topology were added in the revised version described in the next section.

The basic CSMA/CA mechanism relies on the CCA (clear channel assessment) signal reported by the baseband processor. The CCA signal is effective only when the NAV (Net Allocation Vector) timer, loaded from the 16-bit Duration Field of the frames from other stations, is expired. This prevents the current station from intervening in the data exchanges of other stations. Upon the need for frame transmission, a number that is uniformly selected from an exponentially increasing contention window (CW) is loaded into the backoff timer at the first time CCA indicates a clear medium. This timer counts down only when the channel is clear (i.e., CCA = 0). As the timer reduces to zero, the queued frame is transmitted if the medium is again sensed clear.

The DCF also employs different time offsets to prioritize the transmitted frames. In general, control frames, such as RTS, CTS and ACK, should have higher priority for medium usage than the normal data/management frames. Accordingly, a longer *Distributed Inter-Frame Space* (DIFS) offset is added before the execution of the contention procedure for the data/management frames.

In addition to the backoff timer and IFS timer, an ACK timer is used to monitor the status of the previous transmission procedure. Its expiration indicates the occurrence of a failure transmission, and a retransmission may need to be scheduled. For ease of implementation, these three timers were separately designed in our first MAC version. We, however, found that these three timers indeed take turns to be effective. In other words, only one of the three timers counts at a time, and the others always remain idle before the expiration of the currently effective timer. Consequently, in our second edition, they share the same counter circuitry.

Another timer necessary for the MAC controller is the Timing Synchronous function (TSF) timer, which is used to synchronize the procedures for all stations in a basic service set (BSS). It needs to be periodically adjusted according to the recent Beacon frame, contentionally transmitted by one of the stations in an ad hoc BSS.

The RTS/CTS frame exchange procedure is used to improve the system throughput under the occurrence of hidden nodes. The procedure should be used only when the length of the transmitted frame is greater than the RTS threshold. Our design substantiates that this procedure can be handled by the MAC controller without the intervention of the host driver.

According to the standard, only the first fragment of an MSDU (MAC Service Data Unit) is required to execute the CSMA/CA mechanism and RTS/CTS frame exchange procedure; the remaining fragments can be transmitted directly within the SIFS time interval. This is used to uphold the integrity of an MSDU for higher layer protocol stack.

2. Transmission and Reception Finite State Machine

In Fig. 1, where the function diagram of our MAC controller is depicted, the transmission finite state machine (Tx_FSM) implements the DCF and provides necessary interfacing signals to the baseband processor.

Contrary to the Tx_FSM, the Reception FSM processes the received frame based on the information extracted from its header. It should tell apart the data frames from the control frames and

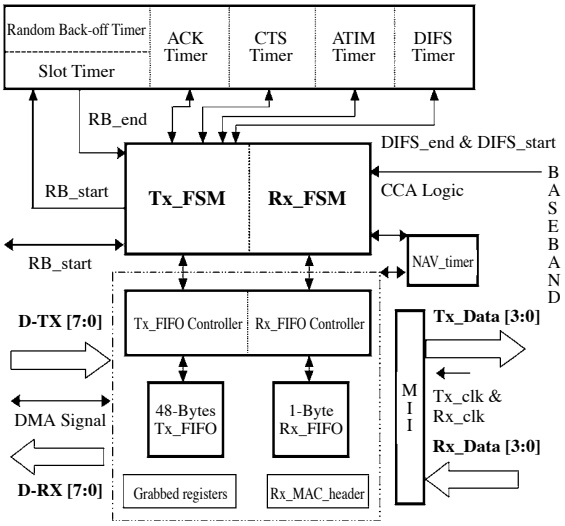

Fig. 1 Architecture of MAC controller

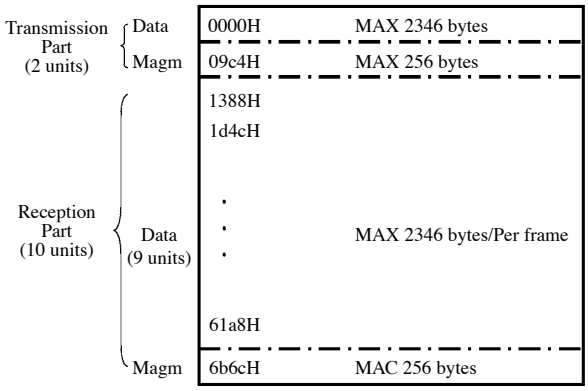

Fig. 2 Storage architecture for external SRAM

management frames, and pass the information to the Tx_FSM to take the proper response. Also examined in this unit is the CRC16 for PLCP header, the protocol version and the uni/multicastness of Address Field 1 (i.e., the destination MAC address). In case an error occurs during these examinations, the reception process as well as the received frame is aborted. At last, a proper status will be placed in the status register, followed by an interrupt to the host driver. The host driver then reads the status register to determine whether a frame is successfully received.

3. Timer Design

In this version, the counters for each timer are separately designed, and run at 44MHz. The IFS timer employs a 12-bit counter, also ticking based on the 44MHz system clock. The ACK timer for MAC-level acknowledgement should measure the transmission time of an ACK frame plus additional 10 μs, which depend on the Signal Field of the PLCP header of the frame being acknowledged. In case the ACK timer expires before the reception of an ACK, an interrupt is issued to the host driver to indicate the failure of previous transmission. The CTS timer is designed similarly to the ACK timer except that it measures the anticipated return time of a CTS frame upon the transmission of an RTS frame. The ATIM timer is used when the station is in an ad hoc network.

The NAV timer is used to carry out the virtual carrier sense scheme. Its initial value is equal to the Duration Field of the received frame in milliseconds. In our design, the countdown process begins after the reception of the last byte of a frame from the baseband processor. Its update is performed when the Duration Field of a newly received frame is greater than the current NAV value.

4. FIFO Controller

The Tx/Rx_FIFO controller in Fig. 1 acts as a data flow arbitrator. It either derives from the baseband processor to the external SRAM for reception, or reads from the external SRAM to the baseband processor for transmission. Both directions follow the 4-bit-wide MII interface, and result in a reduction of the transmission/reception clocks to 250/500 KHz respectively for 1/2 Mbps data rates. Additional implicit signal is implemented to differentiate the current 4-bit input between the least- and most-significant 4 bits. The Tx_FIFO is 48-byte wide, a length to sufficiently store the entire frame header. Since a large FIFO consumes a lot of gate counts, an alternative design has been used in our revision. The Rx_FIFO is only one byte in length, which is sufficient due to the fact that the reception clock is much slower than the SRAM access clock.

5. Bus Interface Unit (BIU) and Data Storage on External SRAM

The bus interface to the host computer follows the PCMCIA 2.01 standard (1992). As the PCMCIA unit is designed solely for on-board verification of the MAC controller, we fix the I/O port addresses to be 280H, 281H and 282H.

The external SRAM is presumed to be 32K×8 in size. In our first trial design, a fixed partition on the external SRAM is adopted for design convenience, where each partition is 2.5 Kbytes in size, which is much larger than 2346 bytes-the largest possible for an IEEE 802.11 data frame. As shown in Fig. 2, two 2.5K byte partition units are respectively dedicated to the transmission of data and management

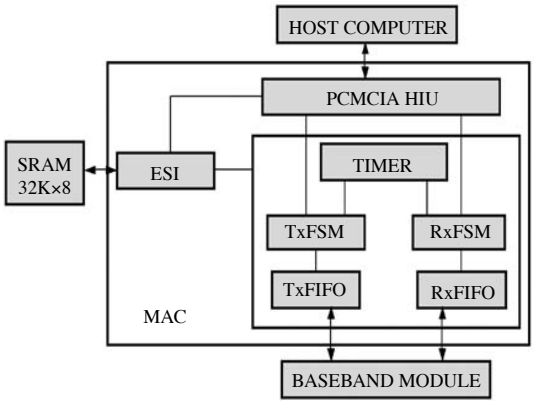

Fig. 3 Architecture of the MAC controller (revised version)

frames. Nine and one fixed-sized partition units are reserved respectively for the reception of data and management frames. In our design, the management frames, owing to their nature, have higher priority than the normal data frames.

6. CRC Check and Gate Count Summary

A CRC unit is designed to perform the generation of transmission CRC32 and the examination of reception CRC32. To adapt to the 4-bit MII interfaces, a parallel CRC architecture is implemented (Pei, 1992). As anticipated, a failure in reception CRC32 check will ignite the deletion of the received frame.

The resultant gate counts for ASIC-implementation and FPGA-implementation are respectively 27, 000 gates (Synopsys) and 4600 logic cells & 1500 flip-flops.

III. ADVANCED DESIGN OF THE MAC CONTROLLER

After the success of trial combinational logic design, we proceed to work on its refinement. Simulations on the previous version only provide us with a half-duplex processing speed of 9 Mbps due to some critical path delays. Other critical issues in the previous design are (1) stick to 44MHz system clock; (2) inefficient use of external SRAM; (3) large gate counts due to the usage of big FIFOs; (4) no control of time-critical management frames; (5) no support to PCF; and (6) no support to infrastructure topology. These issues are resolved in the new MAC edition.

As illustrated in Fig. 3, the new design actually follows the basic structure of the previous design. The baseband interface becomes serial (instead of 4-bit-wide MII parallel interface) in order to comply with commercialized basedband processors such as HFA3861.

Table 1 Gate counts for each unit of the MAC controller (revised version)

UNITS	GATE COUNTS
PCMCIA HIU	4015 gates
DMA/ESI	323 gates
RxFIFO	2116 gates
TxFIFO	968 gates
RxFSM	7834 gates
TxFSM & TIMER	6437 gates

The ESI (external SRAM interface) handles data access to-and-from the PCMCIA HIU (Host interface unit), RxFIFO, TxFIFO and the external SRAM. It arbitrates the data flows of (1) SRAM to host through PCMCIA HIU, (2) host to SRAM through PCMCIA HIU, (3) SRAM to TxFIFO, and (4) RxFIFO to SRAM. The latter two flows have higher priority than the former two. The external SRAM interface now addresses up to 64K(8 to fulfill the buffer need of an access point (AP). To efficiently use the memory space, the received frames are now consecutively stored in the external SRAM, and an indirect and transparent access scheme for the host to the SRAM is implemented. Part of the SRAM space is reserved for the storage of Beacon frame, which is periodically and automatically transmitted by the MAC controller if the station acts as an AP. The synchronization of the TSF timer is now self-updated and maintained by the MAC controller. The host driver (of an AP) can update the content of the Beacon frame whenever necessary.

Due to an effort to synchronize the SRAM access with the baseband transmission, the TxFIFO now only employs a 32-bit register. Also by a complete clock synchronizing design, the MAC controller now adapt to various system clocks, ranged from 11 MHz to 44 MHz. Three management frames, i.e., Beacon, ATIM and Probe Response, are handled without the intervention of the host driver. This can further ease the burden on a host driver.

As aforementioned, the backoff timer, the IFS timer and the ACK/CTS timer now share the same counter so that further reduction of gate counts is rendered. The support to PCF is achieved by a flexible disability of the CSMA/CA mechanism through a register configuration. The resultant gate count for each unit is listed in Table 1.

IV. SYSTEM VERIFICATIONS

The verification and testing of a protocol processor is a significant part of its development. As our target is a hardware MAC controller, the first