RSP: Development of an 802.11 MAC'less Card Controller

Richard L. Lynch

Advisors: Dr. W. Thomas Miller and Kevin J. Karcz

Electrical and Computer Engineering Department

University of New Hampshire

May 11, 2001

Abstract

In the growing world of ubiquitous networking, people often take it for granted that interoperability exists between their network devices and the networks they are connecting to. Consortiums of vendors and international standard bodies are formed to establish standards on how devices should behave if they wish to communicate with each other. However, verifying conformance to these standards is no trivial task. The original IEEE 802.11 specification spent over 400 pages defining how IEEE 802.11 compliant devices should behave.

The Wireless Consortium at the UNH InterOperability Lab tests 802.11 Wireless Local Area Network (WLAN) devices to ensure they comply with the IEEE 802.11 standard. The RSP was developed to add to the consortium's testing capabilities. The RSP has the capability of reacting to wireless traffic in a manner that tests a device's handling of uncommon, but very important conditions. The RSP consists of an 802.11 MAC'less card from Intersil, interfaced to an Altera programmable logic device, which in turn communicates with a PC through a USB microcontroller. The Altera Programmable Logic Device (PLD) was programmed using Verilog.

Table of Contents

1. Introduction

People often take it for granted that interoperability exists between their network devices and the networks they are connecting to. Most consumers would find it very frustrating and annoying if the Ethernet card they purchased from vendor A would not connect to the campus/corporate Ethernet network using equipment from vendor B. These sorts of problems are partially solved through the creation of standards that define how devices should behave. However, ensuring conformance to these standards is no trivial task. The original IEEE 802.11 standard spent over 400 pages describing how wireless LAN should behave. Not all the rules set forth in the standard are necessary for interoperability. Some parts of the standard only come into effect under certain conditions. For instance, the 802.11 exponential backoff algorithm is primarily important under heavy traffic conditions when collisions occur frequently. These portions of the standard cannot be tested by simply turning on a few "ideal" devices and seeing if they interoperate with a device under test (DUT). Often, these portions of the standard require the ability to generate certain uncommon frames or conditions in a controlled manner. The RSP was developed to fill this gap by creating a device capable of reacting to conditions on the wireless medium with carefully crafted frames that exercise a DUT and ensure its conformance to the IEEE 802.11 standard.

2. 802.11 Background

The IEEE 802.11b standard defines the manner by which WLAN devices should communicate in the unlicensed 2.4GHz ISM band at rates up to 11Mbps. 802.11b is based on a Direct Sequence Spread Spectrum PHY with BPSK/QPSK/CCK modulation.

Two network topologies are defined – ad hoc and infrastructure. In ad hoc (peer to peer) mode, each wireless network device communicates with every other device directly. In infrastructure mode, all frames are sent through a central device called an access point (AP, similar to a switch). The AP also provides access to the wired network.

Unlike Ethernet networks, 802.11 is based on Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA). The CSMA/CA mechanism allows wireless devices to reserve the medium using special control frames to minimize wireless collisions. Although this complicates the protocol, it is necessary since wireless transmitters cannot detect collisions. The remaining collisions are detected by the absence of an acknowledgement frame.

Security is provided through authentication and encryption using the Wired Equivalent Privacy (WEP) protocol. WEP encrypts data frames using an RC4 PRNG seeded with a 40/104 bit shared key and a 24 bit initialization vector. RC4 is presently used in Electronic Codebook (ECB) mode. IEEE 802.11e is looking to eliminate some of the attacks discussed by Nikita Borisov, Ian Goldberg, and David Wagner in http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html by expanding the initialization vector to 128 bits and alternately switching to the recently approved Advanced Encryption Standard (AES) algorithm in Offset Codebook (OCB) mode.

More detailed information on 802.11 is included in appendix D.

3. System Level Discussion

As shown in figure 1, the RSP consists of four major modules – an Intersil MAC'less card, an Altera PLD, a USB microcontroller, and a host PC.

The Intersil MAC'less card contains the PRISM II chipset with the Medium Access Controller (MAC) removed. The MAC is normally responsible for forming frames, authenticating and associating with access points, determining when it is permissible to use the wireless medium, etc. By using a MAC'less card, it is possible to break some of the 802.11 rules and generate a variety conditions that would otherwise be cumbersome to create.

The Altera PLD was programmed in Verilog and essentially contains a test MAC. The test MAC is responsible for all of the functionality that would have been handled by a regular MAC. The test MAC is capable of initializing the baseband processor registers, configuring the synthesizers, capturing incoming frames to memory, reading outgoing frames from memory and feeding them to the baseband processor, and many other time critical operations. The Altera PLD is also responsible for managing the transmit and receive SRAM.

The USB microcontroller contains an enhanced 8051 microcontroller and USB interface. It is responsible for periodically polling the Altera PLD's state and relaying commands from the host PC to the Altera PLD.

The host PC is responsible for all high level functionality. It is responsible for converting the RSP scripting language from text form into binary form, loading frames into the transmit SRAM, retrieving frames from the receive SRAM, issuing configuration commands to the Altera PLD, etc.

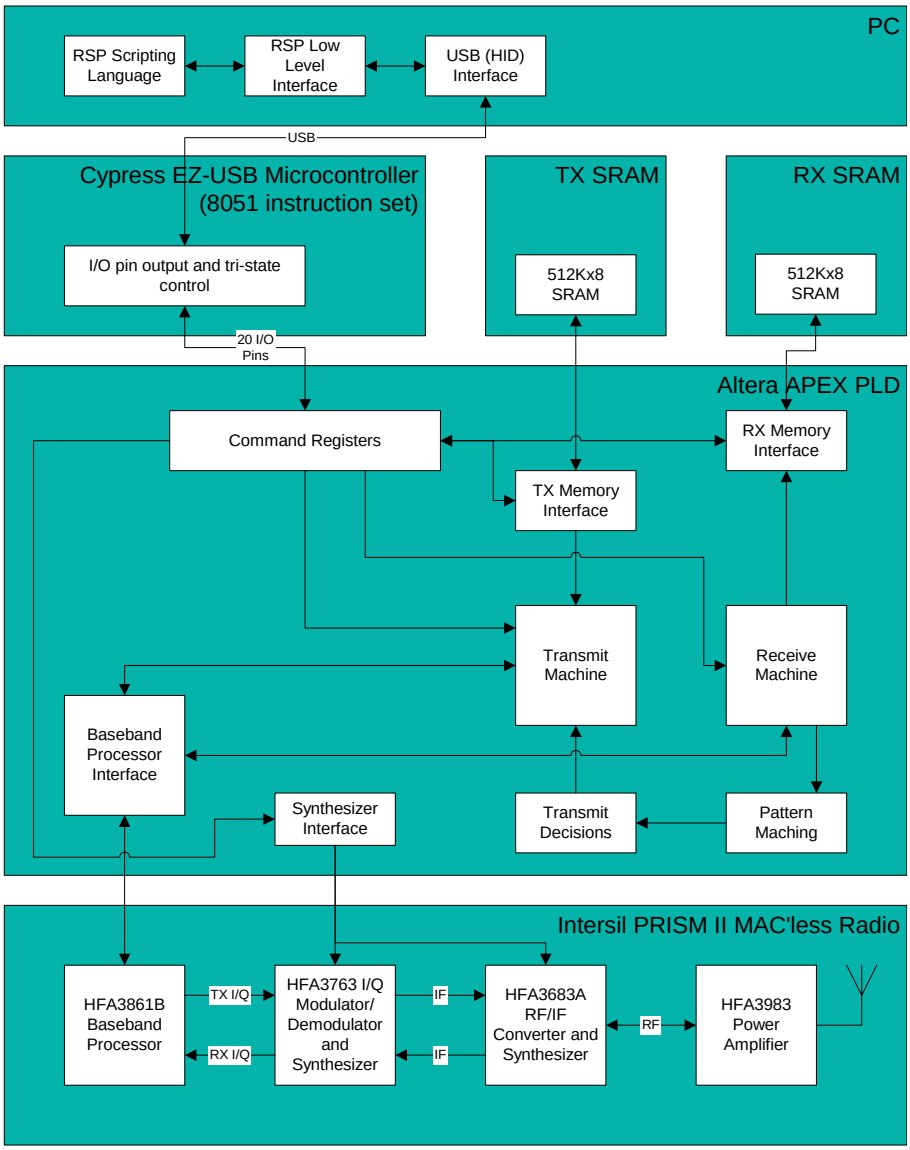


Figure 1 – System Level Block Diagram

4.

Intersil MAC'less Card Discussion

The Intersil MAC'less card contains the PRISM II chipset with the MAC removed. The MAC functionality missing from the Intersil card is included in the Altera PLD. By creating a custom MAC, it became possible to transmit arbitrary frames at precise adjustable times.

The operation of the MAC'less card was initially characterized using an HP16500B Logic Analyzer System. Labview code was developed to program the logic analyzer and pattern generator to perform simple functions, such as writing/reading baseband registers, initializing the synthesizer, transmitting a single frame repeatedly, configuring the logic analyzer triggers to capture a frame, etc. This was an important stage of the RSP's development as documentation on the Intersil card was scarce and plagued with gaps and inaccuracies. However, the logic analyzer could not be used for actual testing as it was slow to program and lacked the computational power necessary to carry out most of the MAC tests.

5. Altera PLD Discussion

Kevin J. Karcz developed a PCB board for the project that included an Altera PLD. The Altera PLD ultimately became a test MAC – capable of responding to a variety of conditions with either a frame transmission after a precise delay or generating an external trigger. Significant time was spent dealing with quirks in the Intersil card, many of which are listed in appendix B.

The Altera PLD was programmed in Verilog. Currently, three Verilog modules exist – transmit_signal, patternmatch, and the main module. Each transmit_signal module decides when a particular frame is ready for transmission. Each patternmatch module searches for frames conforming to a certain criteria (e.g. a 2Mbps acknowledgement frame). The main module instantiates the other modules and is responsible for interfaces to the USB microcontroller and Intersil card.

Up to 16 transmit_signal modules can be connected in a bus style structure and configured from the main module. Each transmit_signal module has an identifier coded into it that is used by the main module to select it. When selected, a transmit_module can either be configured, or return its configuration information. Some of the configuration information is necessary for

transmission, such as starting address of the frame in transmit memory. Other configuration information is used to determine when to generate the "want to transmit" signal (e.g. transmission interval, condition to transmit on, etc.).

The patternmatch modules are also arranged into a bus structure, but only have a single output – "condition met". "Condition met" indicates whether or not the configured criteria have been met. The criteria include searching for a frame with a specific rate, a specific length, containing a specific string, or a counter register containing a specific value. When the outputs from the patternmatch modules are grouped together, a status word is formed. This status word is used by the transmit modules to decide if any of them wish to transmit. In the future, counters will use the status word to determine if they should increment.

The main module instantiates all the other modules and is responsible for controlling input/output pins on the PLD. Several code blocks exist in the main module. Noteworthy ones are listed below:

- Timestamp Module – keeps track of the time since power up. Used for timestamping incoming frames and for generating a slow clock for debugging purposes.
- Receive Memory Controller – arbitrates access to the receive SRAM. During frame reception, a special flag is set, giving the frame reception code block priority to the receive SRAM over the receive SRAM control code block.
- Receive SRAM Control – Responsible for testing the receive SRAM, and fetching received frames from the receive SRAM for the host PC. Presently, receive SRAM is read back in 15 byte blocks.
- Frame Reception Code Block – Responsible for determining when the end of a frame has passed, storing received frames to SRAM, and forming/writing the header blocks describing the received frames. 64KB at the beginning of the receive memory was allocated for 2048 32 octet header blocks describing each frame received. The remaining memory is filled in a circular fashion with the received frames themselves.
- Transmit Memory Controller – arbitrates access to the transmit SRAM. During frame transmission, a special flag is set, giving the frame transmission code block priority to the transmit SRAM over the transmit SRAM control code block.
- Transmit SRAM Control – Responsible for testing the transmit SRAM and loading frames to be sent into SRAM.

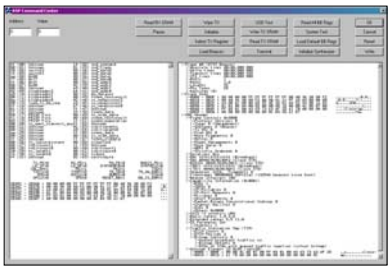
- Frame Transmission Code Block – Responsible for transmitting a frame upon receiving a signal from a transmit_signal module. The code block contains sub-blocks capable of enabling power to the different parts of the Intersil card with the proper timing relationships, fetching a new byte to transmit every 8 TX_CLK clock cycles, performing parallel to serial conversion of the data, and programming the baseband registers associated with the PLCP header.
- IF/RF Synthesizer Block – Responsible for configuring the IF/RF synthesizers with the values specified by the host PC.
- Debugging Registers Module – Exposes the internal state of the PLD to the USB microcontroller in the form of 128 registers.
- Command Block – Reads 33 byte commands from the USB microcontroller 8 bits at a time into a giant shift register.
- Pattern Match Configuration Code Block – Configures the patternmatch modules with information from the host PC
- Transmit Signal Configuration Code Block – Configures the transmit_signal modules with information from the host PC

6. USB Microcontroller Discussion

The Cypress EZ-USB Microcontroller provides a convenient interface to the PC. It contains an enhanced Intel 8051 microcontroller. The microcontroller has no non-volatile memory, but instead has the ability to "reenumerate" itself after power-up. Upon power-up, the microcontroller enumerates itself with the host PC using a hard coded vendor and product identification. The host PC then loads the driver associated with the vendor and product ID, in this case, a special bootstrap driver capable of loading the real microcontroller code into the microcontroller's RAM. Once this process is complete, the microcontroller performs a virtual disconnect, reconnects with the new vendor and product ID, and the host PC in turn loads the real driver for the device. To reduce the complexity of the project, the generic human interface device (HID) drivers from Microsoft were used with the RSP. Classic HID devices directly interact with people and include keyboards, mice, etc. However, the class is flexible and can be used for any low bandwidth application.

Sample HID code provided with the EZ-USB microcontroller was modified to periodically poll the Altera PLD debugging registers, form a "report" out of these, and send the report to the PC. It is also responsible for loading 33 byte commands from the host PC into the Altera PLD.

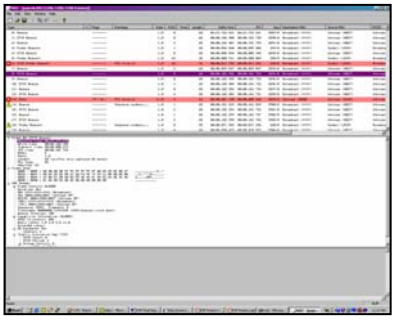
7. Host PC Software Discussion



The host PC software consists of three layers – a USB interface layer, an RSP interface layer, and the scripting layer. The USB interface layer provides a layer of abstraction between the RSP interface layer and the USB HID drivers. The RSP interface layer provides convenient access to the RSP functionality without worrying about how specific commands operate within the Altera PLD. Finally, the scripting language will convert text file descriptions of tests into binary data suitable for loading into the RSP. This layer has not been completed yet.

The lower two layers are presently implemented in the RSP Command Console. Currently, the console only provides debugging information, but ultimately, the interface will be cleaned up and be easy to use. An interface to NSI2 has been created in the command console. NSI2 is an 802.11 protocol analyzer I developed that is capable of passively locating implementation problems in wireless devices, displaying traces of wireless activity in various forms, including plot and text summary, and many other useful functions. By combining RSP with NSI2, a "sniffer" can be created to display all of the traffic passing over the wireless medium.

8. Results/Verification



All of the basic RSP functionality is in place. The RSP can receive frames and relay them back to the PC (sniffer mode), transmit frames periodically or in response to either an external signal, and recognize patterns in incoming frames. The RSP cannot respond to an incoming frame with a frame transmission, but the code to perform this is trivial. The current primary problem is the power level of frames, transmitted by the Intersil card, are below the 802.11 minimum specified receiver sensitivity level. As such, the RSP's transmission range is very limited. Bit error rate for incoming frames is also high, and the PLCP header of most frames (typically >90%) is

received incorrectly, causing many frames to be dropped all together. These problems seem to be caused by either a defect or damage to the Intersil MAC'less card.

Verification of functionality was accomplished through the use of a logic analyzer to monitor the behavior of the Verilog code, an 802.11 "sniffer" from Network Associates to verify frames were being transmitted correctly, and an Agilent 89600 vector signal analyzer.

9. Conclusion

All of the basic objectives of the RSP have been met. The remaining objectives will be completed over the summer, but are dependent on the problems with the Intersil card being resolved. Alternately, there is the possibility of replacing the Intersil MAC'less card with a Sharewave MAC'less card, but that is up in the air and a difficult path. In any case, the RSP will progress and ultimately aid in 802.11 MAC testing at the IOL.

Appendix A – Intersil MAC'less Card Pinout

<table>
<tr><th>Pin Name</th><th>Direction (relative to MAC'less Card)</th><th>Pin Number</th><th>Description</th></tr>
<tr><td>C_TX_PE</td><td>I</td><td>29</td><td>Baseband transmit enable. Should be enabled prior to C_PA_PE, and disabled after C_PA_PE is disabled.</td></tr>
<tr><td>C_RX_PE</td><td>I</td><td>27</td><td>Baseband receive enable. Must be cycled each time a frame is received, otherwise, the baseband processor will keep demodulating past the end of the frame.</td></tr>
<tr><td>C_PA_PE</td><td>I</td><td>25</td><td>Power amplifier enable. Boosts power to transmission level.</td></tr>
<tr><td>RADIO_PE</td><td>I</td><td>24</td><td>Enables power to all chips. Should be kept high during operation of card</td></tr>
<tr><td>LE_IF</td><td>I</td><td>21</td><td>Latch IF synthesizer value. The rising edge of this line causes the data from SYNTHDATA to get latched into the IF synthesizer.</td></tr>
<tr><td>LE_RF</td><td>I</td><td>43</td><td>Latch RF synthesizer value. The rising edge of this line causes the data from SYNTHDATA to get latched into the RF synthesizer</td></tr>
<tr><td>SYNTHDATA</td><td>I</td><td>23</td><td>Synthesizer data in. Data is latched on the rising edge of SYNTHCLK.</td></tr>
<tr><td>SYNTHCLK</td><td>I</td><td>22</td><td>Synthesizer clock in.</td></tr>
<tr><td>CAL_EN</td><td>I</td><td>19</td><td>The purpose of this pin is uncertain.</td></tr>
<tr><td>PE2/PE1</td><td>I</td><td>PE2 = 11
PE1 = 12</td><td>Power enable control pins to the IF synthesizer.
<table>
<tr><th>PE1</th><th>PE2</th><th>Status</th></tr>
<tr><td>0</td><td>0</td><td>Power down state, PLL registers in save mode, inactive PLL</td></tr>
<tr><td>1</td><td>1</td><td>Receive state, active PLL</td></tr>
<tr><td>1</td><td>0</td><td>Transmit state, active PLL</td></tr>
<tr><td>0</td><td>1</td><td>Inactive transmit and receive state, active PLL</td></tr>
</table></td></tr>
<tr><td>TXD</td><td>I</td><td>28</td><td>Transmit data. Data is clocked in on the rising edge of the TXC. First rising edge is discarded.</td></tr>
<tr><td>TXCLK</td><td>O</td><td>33</td><td>Transmit clock.</td></tr>
<tr><td>TXRDY</td><td>O</td><td>67</td><td>Ready to transmit data. Indicates when the preamble has been transmitted, and the baseband processor is ready to receive the MPDU. Is deasserted after TX_PE goes low.</td></tr>
<tr><td>TR_SW</td><td>I</td><td>40</td><td>Transmit/Receive switch. 0 = RX, 1 = TX</td></tr>
<tr><td>TR_SW_BAR</td><td>I</td><td>42</td><td>Active low Transmit/Receive switch. 0 = TX, 1 = RX</td></tr>
<tr><td>RXD</td><td>O</td><td>63</td><td>Receive data. Data should be latched on the rising</td></tr>
</table>

			edge of RXCLK. Data valid until about 40ns after rising edge.
RXCLK	O	36	Receive clock
MDRDY	O	59	Indicates when a valid preamble/complete PLCP header has been received (adjustable via configuration register 10).
CCA	O	62	Clear Channel Assessment
SPCLK	I	20	Baseband processor serial clock
SPD	I/O	30	Baseband processor serial data. When writing registers, the MSB of the address must be set. When reading registers, the MSB of the address must be zero.
RESET_BB		26	Active low reset baseband processor. When active, transmit and receive functions are disabled. Does not alter configuration registers.
BB_CS_BAR		14	Enables the baseband processor serial port. Active low.
Vcc		17, 51	+3.3V power.
GND		1, 34, 35, 68	Ground

Appendix B – Intersil Card Unexplained Behavior/Observations

1. Frame reception is poor (high bit error rate among the small percentage of frames received) if the Altera PLD does not periodically transmit frames. Decreasing the period of transmission increases the percentage of frames received.
2. Periodically toggling the RESET_BB seems to improve reception.
3. Periodically writing 0x9b then 0x1b to baseband register 0x16 seems to improve reception.
4. Transmission power is always low, no matter what setting is used for baseband register 0x3e.
5. A plot of the signal strength vs. time has a square wave appearance with a 10us period. Both MAC'less cards exhibit this behavior. Normal devices do not exhibit this behavior.
6. It may be necessary to reset the baseband processor after receiving a frame with an invalid PLCP CRC-16 (this is already being done).

Appendix C – Synthesizer Programming

Example calculations

IF:
F_{ref} = (44MHz reference clock) / 2 = 22MHz
R = 44
F_{ref}/R = 500kHz steps for A and B
A = 12
B = 46
P = 16
$F_{IF} = (P*B+A)*(F_{ref}/R)$ = 374 MHz

RF:
F_{ref} = 44MHz reference clock
R = 44
F_{ref}/R = 1MHz steps for A and B
A = 22
B = 63
P = 32
$F_{RF} = (P*B+A)*(F_{ref}/R)$ = 2038 MHz

2038MHz + 374MHz = 2412MHz – channel 1

Standard Values

		IF					RF				
Channel	**Frequency**	**A**	**B**	**P**	**R**	**F_{IF} (MHz)**	**A**	**B**	**P**	**R**	**F_{RF} (MHz)**
1	2412	12	46	16	44	374	22	63	32	44	2038
2	2417	12	46	16	44	374	27	63	32	44	2043
3	2422	12	46	16	44	374	0	64	32	44	2048
4	2427	12	46	16	44	374	5	64	32	44	2053
5	2432	12	46	16	44	374	10	64	32	44	2058
6	2437	12	46	16	44	374	15	64	32	44	2063
7	2442	12	46	16	44	374	20	64	32	44	2068
8	2447	12	46	16	44	374	25	64	32	44	2073
9	2452	12	46	16	44	374	30	64	32	44	2078
10	2457	12	46	16	44	374	3	65	32	44	2083
11	2462	12	46	16	44	374	8	65	32	44	2088
12 (Europe)	2467	12	46	16	44	374	13	65	32	44	2093
13 (Europe)	2472	12	46	16	44	374	18	65	32	44	2098

Appendix D – Informational Handouts

Attached are handouts from the presentation that describe wireless networks in general and 802.11 in specific.

- *What is a Wireless LAN?* by Proxim, Inc.
  http://www.proxim.com/wireless/whiteppr/whatwlan.pdf
- *IEEE 802.11 Tutorial* by Jim Zyren and Al Petrick
  http://www.wirelessethernet.com/downloads/IEEE_80211_Primer.pdf
- *What is IEEE 802.11 Compliance?* by Lucent Technologies, Inc.
  ftp://ftp.orinocowireless.com/pub/docs/IEEE/BULLETIN/SALES/sb_80211.pdf