A Proposal of Hardware Channel Bonding for IEEE802.11n Wireless Network Using Raspberry Pi

Rahardhita Widyatra Sudibyo and Nobuo Funabiki
Department of Electrical and Communication Engineering
Okayama University, Okayama, Japan
Email: funabiki@okayama-u.ac.jp

Wen-Chung Kao
Department of Electrical Engineering
National Taiwan Normal University, Taipei, Taiwan
Email: jungkao@ntnu.edu.tw

***Abstract*—*Raspberry Pi* has become popular around the world as a small-size, low-cost, low-power, and high-performance computing device. It can be used as a software *access-point* for IEEE802.11n wireless local-area networks. Unfortunately, only the *software channel bonding* is available where two independent 20MHz channels are used together for each link. In this paper, we present the configuration of *Raspberry Pi* for the *hardware channel bonding* to increase the capacity with 40MHz bandwidth. Because the built-in wireless NIC does not support it, the external adapter with USB3.0 is used together. To verify the performance, we conducted throughput measurements in two scenarios.**

I. Introduction

The data transmission speed of a *wireless access-points (AP)* is contributing factor in determining the performance of a *IEEE802.11 wireless local-area network (WLAN)*. In consequence, several technologies have been proposed, and this paper will focus on the *channel bonding* [1].

Raspberry Pi has become dominant around the world as a small-size, low-cost, low-power, and high-performance computing device. It can be used as a software AP for IEEE802.11n WLAN using *hostapd* [2], since it adopts the wireless NIC (network interface card) for 11n and Linux OS. Unfortunately, this software AP can realize the *software channel bonding* where two independent 20MHz channels are used together to produce a communication link between an AP and a host. The real *hardware channel bonding* uses 40MHz bandwidth for one channel by bonding two adjacent 20MHz channels.

Orthogonal Frequency-Division Multiplexing (OFDM) in IEEE802.11n uses 52 *subcarriers* for data transmissions for the 20MHz channel and 110 subcarriers for the 40MHz channel. Thus, the latter can increase the capacity compared with the former [3]. As a result of larger transmission capacity, the *guard interval* will be reduced while the *frame aggregation* will be increased. Therefore, the hardware channel bonding can provide higher speed than the software channel bonding.

In this paper, we present the configuration of *Raspberry Pi* [4] to activate the hardware channel bonding. Because the wireless NIC built in *Raspberry Pi* does not support it, we introduce the external wireless NIC adapter with USB3.0.

To verify the performance, we conducted throughput measurements in two scenarios, using this *Raspberry Pi* with the software/hardware channel bonding and a commercial 11n AP. In the first scenario, the distance between the AP and the host is changed from $0m$ to $30m$ in an outdoor environment, where the throughput of the proposed AP is similar to that by the commercial AP and better than the software channel bonding. In the second scenario, the AP is fixed at one place in the corridor and the host is located at different places in different rooms in an indoor environment, where the throughput is much smaller than that of the commercial AP when the host is distant from the AP.

II. Configuration for Hardware Channel Bonding

In this section, we present the configuration procedure for *Raspberry Pi 3* AP with the hardware channel bonding.

1) *hostapd* and *bridge-utils* are installed from *Raspbian Repository* by using the command: `sudo apt-get install hostapd bridge-utils`.
2) *hostapd* is rebuilt to support the software channel bonding by:
   a) *deb-src* in */etc/apt/sources.list* is activated and updated by using the command `sudo apt-get update`.
   b) The source code of *hostapd* is obtained and it is built by using the command `sudo apt-get source hostapd` and `sudo apt-get build-dep hostapd`.
   c) *hw_features.c* is modified and rebuilt
   d) *hostapd* is reinstalled using *dpkg* command.
3) The configuration file in *hostapd.conf* is created where the configuration bridge = br0; hw_mode = g; ieee80211n = 1; obss_interval = 1; ht_capab = [HT40+] [SHORT-GI-20] [SHORT-GI-40] [DSSS_CCK-40] [MAX-AMSDU-7935].
4) The interface is edited to create a bridge from the wireless adapter to the Ethernet adapter in */etc/network/interfaces* file. Static IP address is set.
5) *hostapd* is initiated by the command `sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf`.

III. Throughput Measurements

In this section, we describe throughput measurement results.

A. Measurement Setup

The devices in Table I were used in measurements on the third floor in Engineering Building #2 at Okayama University.

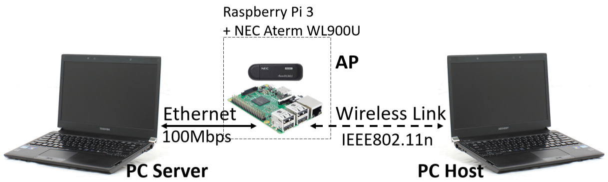

Fig. 1: Throughput measurement system.

TABLE I: Wireless adapter and PC specification.

model	wireless chipset
Raspberry Pi 3	Broadcom BCM43438
NEC WL900U USB3.0	Realtek RTL8812AU
NEC WG2600HP	Qualcomm
model	Dynabook R731/B
processor	Intel Core i5-2520M @2.5Ghz
memory size	4Gb DDR3 1333Mhz
OS	Linux Ubuntu 14.04
software	Iperf 2.0.5

In the measurement system in Figure 1, the AP was located at $135cm$ height from the ground and the host PC was at $70cm$ height. After starting *hostapd* in *Raspberry Pi 3*, *Wifi Analyzer* was used to examine the frequency band usages at 2.4GHz from other devices in the field. Then, the host PC was associated with the AP, and *iperf* was used to generate TCP traffics in both directions with the $477KB$ TCP window size and the $8KB$ buffer size.

B. Measurement Results at Different Distances

First, the throughput was measured when the distance between the AP and the host PC was increased from $0m$ to $30m$ with $5m$ interval in the corridor. Figure 3 shows the results, where the throughput by the proposal is much larger than that by the conventional one at any distance and is similar to that by the commercial AP. In this case, the corridor is similar to a tunnel, and the wireless signal propagates without a big loss. As a result, the throughput by any AP was not decreased so much as the distance increased, and the throughput by *Raspberry Pi* with the hardware channel bonding is nearly the same as that by the commercial AP.

C. Measurement Results at Different Rooms

Then, the throughput was measured when the host PC was located in 14 places in Figure 2. Figure 4 shows the results, where the throughput by the proposal is generally larger than the conventional but is smaller than the commercial AP. In this case, the wall between the corridor and each room becomes the obstacle of the signal propagation where the wireless signal suffers from a big loss. In consequence, the throughput by any AP device becomes smaller at most rooms. As well, the throughput of *Raspberry Pi* will become smaller when the signal needs to pass through two or more walls. It indicates that the current *Raspberry Pi* AP may not be suitable in a field which contains a plenty of obstacles.

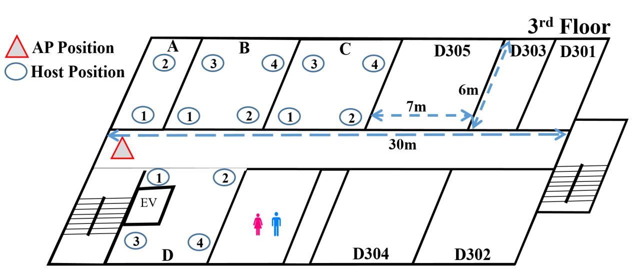

Fig. 2: Throughput measurement field.

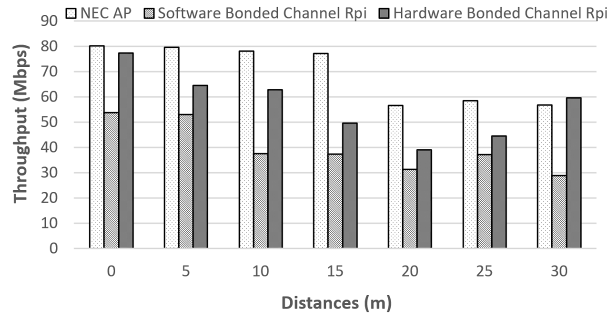

Fig. 3: Throughput measurement results at different distances.

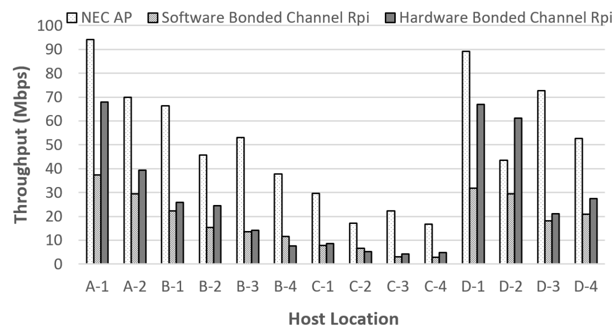

Fig. 4: Throughput measurement results at different rooms.

IV. Conclusion

This paper presented a *Raspberry Pi* configuration using USB3.0 wireless NIC adapter for the AP in IEEE802.11n WLANs to use the hardware channel bonding. Future works will include further throughput measurements with different adapters in indoor environments.

Acknowledgments

This work is partially supported by JSPS KAKENHI (16K00127).

References

[1] L. Deek, E. G. Villegas, E. Belding, S. Lee, and K. Almeroth, "Intelligent channel bonding in 802.11n WLANs," IEEE Tran. Mob. Comput., vol. 13, no. 6, pp. 1242-1254, June 2014.
[2] S. K. Debnath, N. Funabiki, K. S. Lwin, Md. S. A. Mamun, R. W. Sudibyo and S. Huda, "Raspberry Pi configuration for access-point and its throughput measurements in IEEE802.11n wireless networks," IEICE Tech. Rep., NS2016-137, pp. 101-106, December 2016.
[3] National Instrument, Introduction to wireless LAN measurements from 802.11a to 802.11ac.
[4] Raspberry Pi 3 Model B, https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.