HW/SW Co-Design of an Adaptive Frequency Decision in the Bluetooth Wireless Network

Sangook Moon, *Member, KIMICS*

Abstract— In IEEE 802.15.1 (Bluetooth) Ad-hoc networks, the frequency is resolved by the specific part of the digits of the Device clock and the Bluetooth address of the Master device in a given piconet. The piconet performs a fast frequency hopping scheme over 79 carriers of 1-MHz bandwidth. Since there is no coordination between different piconets, packet collisions may occur if two piconets are located near one another. In this paper, we proposed a software/hardware co-design of an adaptive frequency decision mechanism so that more than two different kinds of wireless devices can stay connected without frequency collision. Suggested method was implemented with C program and HDL (Hardware Description Language) and automatically synthesized and laid out. The adaptive frequency hopping circuit was implemented in a prototype and showed its operation at 24MHz correctly.

Index Terms— Bluetooth, frequency hopping, wireless, HDL

I. INTRODUCTION

IN the upcoming era of the digital convergence, radio-device-related technology is emerging as the essence of IT industry, such as wireless LAN or Bluetooth.

Bluetooth is attractive in that the service provides a fundamental and ceaseless way to connect people whenever and wherever they want by connecting the information-communication devices with each other, consuming far less electrical power compared with the wireless LAN. The Bluetooth SIG (Special Interest Group) is working on improving specifications and making effort to develop more convenient consumer products worldwide. The power consumption of the Bluetooth devices is categorized into 3 parts, as in Table 1, depending on which class the device is working.

One of the most important parts of Bluetooth wireless communication is to make hopping frequency sequences. Basic pattern is to spreading out 79 frequencies in the ISM band in the pseudo-random fashion in the shape of two hopping trains. However, the frequency collision problem may happen if there are more than two wireless devices using the same frequency within the shared network area [1] (Figure 1). There have been studies to avoid the frequency collision and one of which is that the device saves the history of the collision frequency and not using it in the next hopping trial [2][3].

In this paper, we propose a method of adaptive frequency hopping, which can prevent the devices from colliding with the same hop frequency. We elaborate the device firmware by saving the collision frequency in the special purpose registers. Also, we considered proper hardware-software trade-off in implementing the adaptive frequency hopping circuit.

Table 1. Bluetooth power consumption class range.

Class	Maximum Permitted Power (mW/dBm)	Range (approximate)
Class 1	100 mW (20 dBm)	~100 meters
Class 2	2.5 mW (4 dBm)	~10 meters
Class 3	1 mW (0 dBm)	~1 meter

II. BLUETOOTH BASEBAND

Bluetooth was originally designed as a cable replacement technology aimed at providing effortless

Manuscript received December 31, 2008 ; revised February 27, 2009. Sangook Moon is with the Department of Electronic Engineering, Mokwon University, Daejeon, 302-318, Korea (Tel: +82-42-829-7637, Fax: +82-42-823-8506, Email: smoon@mokwon.ac.kr)

wireless connectivity for consumer devices in an *ad hoc* fashion. In order to allow for deployment almost worldwide, the Bluetooth Special Interest Group (SIG) placed the technology in the unlicensed industrial, scientific, and medical (ISM) band at 2.4 GHz. By designing a comparably straightforward system, the designers of Bluetooth intended for it to have widespread use.

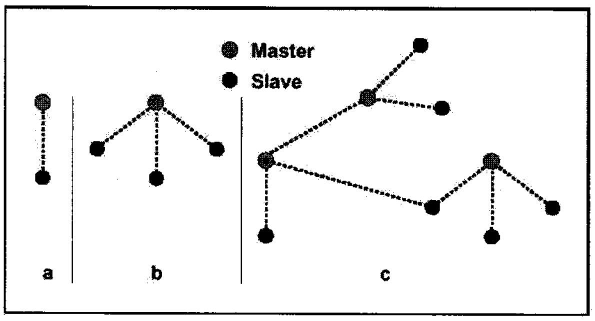


Fig 1. Bluetooth physical communication channel network diagram

Bluetooth networks are organized into "piconets" in which a "master" unit coordinates the traffic to and from up to seven active "slave" units. The master unit originates the request for a connection setup. Within a single piconet, the various slave units can only communicate with each other via the master. Nevertheless, every Bluetooth unit can be a member of up to four different piconets simultaneously (though it can be master in only one of them). A formation in which several piconets are interlinked in such a manner is called a scatternet.

Figure 2 shows the Bluetooth stack layer diagram. Interested area is the lower layer, which is bound with the Host Controller Interface. Figure 3 shows the general block diagram of Bluetooth baseband. The RF (Radio Frequency) module, which is placed in the right-side of the figure, modulates the frequencies from 2.4GHz ISM band to the intermediate frequencies. The data received by the RF module are de-skewed and re-sampled as the exact 1MHz sampled data and given to the low-pass filter to remove the possible noise, later to be passed into the baseband block.

Data are divided into 64-bit blocks, the head of which begins with '1010' or '0101', depending on the type of the packet. Serially injected data are transformed to parallel data and separated into header and payload parts. Header and payload packets are examined in the error correction blocks respectively. FEC (Forward Error Check) scheme is used to examine the packets by the Bluetooth specification [4]. There are two kinds of Bluetooth packet, SCO (Synchronous Connection Oriented) and ACL (Asynchronous Connection oriented Link). SCO packets are used in applications with the voice communication, which needs real-time data slots. ACL packets are used to transfer timely-flexible data [5][6]. The packet consists of three parts, or access code, header, and payload. Figure 4 shows the organization of the Bluetooth packet format.

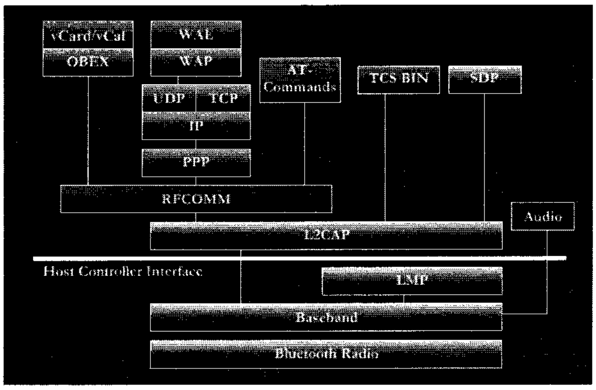


Fig 2. Bluetooth protocol stack diagram

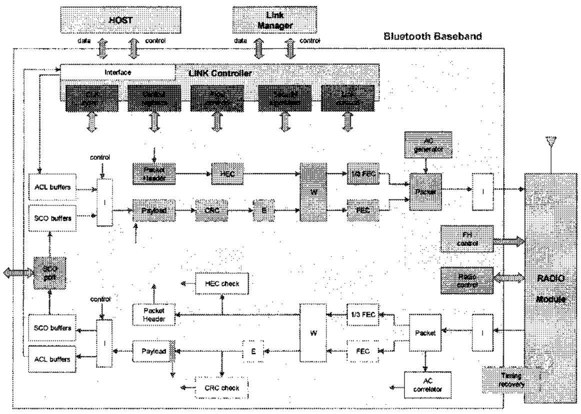


Fig 3. Bluetooth baseband block diagram

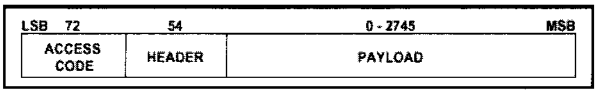


Fig 4. Bluetooth packet format

III. ADAPTIVE FREQUENCY HOPPING

In the Bluetooth baseband, the hopping frequency is resolved according to the rule given in Table 2, depending upon the operation modes, including *page*, *inquiry*, *page scan*, *inquiry scan*, *page response*, *inquiry response*, and *connection*. Figure 5 shows the packet connection flow when two Bluetooth units try to establish the connection. However, in this scheme, there might be chances that another kind of wireless devices which coincidently use the same frequency as the shared one between two Bluetooth units, so that interference could happen.

With the background knowledge of how hop sequences are generated as in Figure 6, of which timing diagram is shown in Figure 7, the proposed adaptive frequency hopping scheme follows the mechanism showed in Figure 8. All of the terms such as UAP/LAP, f_k, PERM5, F', Y2, E, and AFH are defined in the Bluetooth specification version 2.0.

The proposed method operates as follows. At first, normally calculated hopping frequency is selected. Before the selection of the frequency, the frequency is compared with the ones which made collisions. If the frequency is identified to be the same as one in the search table, we designed the module and the firmware to have two options. First, the hardware *remapper* re-maps the frequency hopping table. Second, the programmer can choose a specific hopping frequency so that the devices should not interfere with each other.

The main idea of re-mapping algorithm is that we save the pre-calculated PERM5 output value, the value of F, Y2, and E in a special-purpose register in the baseband processor unit. Also, we designed an efficient hardware dedicated to calculate modulo N operation, where N is a variable number.

Table 2. Bluetooth hopping frequency calculation

	Page scan/ Inquiry scan	Page/Inquiry	Page response (master/slave)	Connection state
X	$CLKN_{15-12}$	$Xp^{(23)}_{3-0}/Xi^{(23)}_{3-0}$	$Xprm^{(23)}_{3-0}/Xprs^{(23)}_{3-0}$	CLK_{5-2}
Y1	0	$CLKE_1/CLKN_1$	$CLKE_1/CLKN_1$	CLK_1
Y2	0	$16 \times CLKE_1/$ $16 \times CLKN_1$	$16 \times CLKE_1/$ $16 \times CLKN_1$	$16 \times CLK_1$
A	A_{27-23}	A_{27-23}	A_{27-23}	$A_{27-23} \oplus CLK_{25-21}$
B	A_{22-19}	A_{22-19}	A_{22-19}	A_{22-19}
C	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0} \oplus CLK_{20-16}$
D	A_{18-10}	A_{18-10}	A_{18-10}	$A_{18-10} \oplus CLK_{15-7}$
E	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$
F	0	0	0	$8 \times CLK_{27-6} \bmod 23$

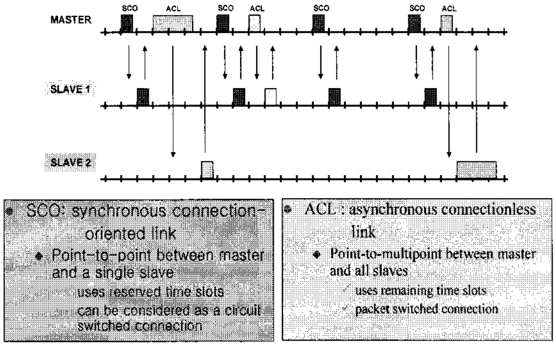


Fig 5. Bluetooth packet communication

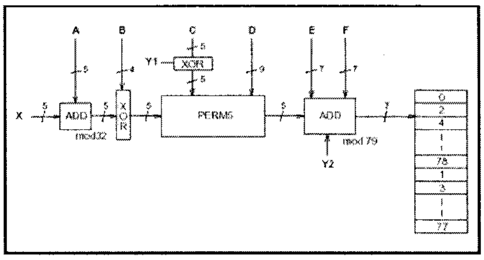


Fig 6. Block diagram of the basic hop selection kernel for the hop system

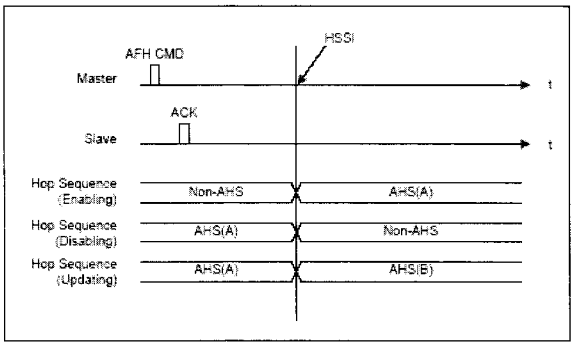


Fig 7. Normal hop sequence switching timing diagram

IV. RESULT AND EVALUATION

Since there is not any coordination between different piconets, packet collisions may occur if two piconets are located near one another. Frequency hopping method minimizes this collision effect as well as to cope with the fact that frequencies used by other devices on the radio channel can vary significantly over the bandwidth of the 2.4-GHz ISM band. The scheme hops over 79 carries of 1-MHz bandwidth each. The maximum hopping frequency of this scheme is set at 1.6 kHz (corresponding to the slot length of 625 s) and the hop sequence used by each individual piconet is derived from the unique address of its master using Table 2

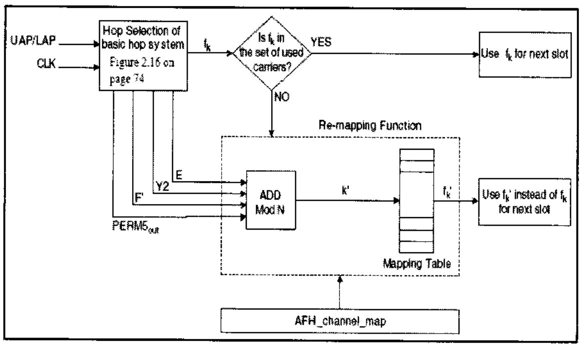


Fig 8. Adaptive frequency hopping mechanism concept.

We implemented two different versions of Bluetooth single (with integrated RF module) chip, one with the ordinary frequency hopping mechanism, the other with the proposed adaptive frequency hopping unit.

The test without any adaptive frequency adjustment unit showed interference or collision between Bluetooth devices with another Bluetooth devices or wireless LANs when the devices tried to hop to the same frequency. However, with the proposed adaptive frequency hopping unit, the frequency collision didn't happen any more since by manipulating the code in the firmware, either the Bluetooth devices automatically avoided the possible collision frequency or we could set the value of the frequency so that no collision happened. The firmware was described with ARM-7 Assembly language in order to reduce the code size.

To make efficient use of the proposed adaptive frequency hopping mechanism, we added special register groups in the reserved memory map region for Baseband, of which the base address starts from 0x80010000, named AFH_map0 through AFH_map4, 16bits each.

Figure 9 shows the timing diagram related with the adaptive frequency hopping in the baseband unit between the hardware control signals and firmware input parameters. *INT*, *AFH_MAP_Table*, *CPU parameters*, and *enable* signal vectors were used as inputs, while *Hop frequency* and *Frequency_Out* signal vectors came out as the result values. We utilized the periodical interrupt signal (*INT*) to balance the timing gap between the microprocessor CPU cycle and the baseband hardware unit. The value of *AFH_MAP_Table* signal vectors can be manipulated by users or the program itself. Whether it should be manipulated by users or automatically in the program is determined in the firmware.

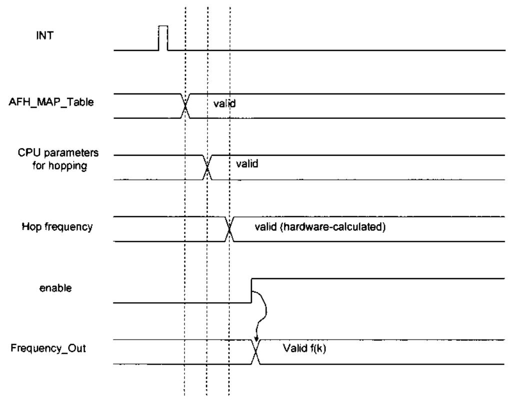


Fig 9. Adaptive frequency hopping timing diagram

In the designed adaptive frequency hopping part, only the variable modulo N calculation block is implemented with hardware, the other implemented with software (firmware) so that we can cope with sudden possible problems with flexibility. The hardware-described circuit was automatically synthesized with Synopsis, and the layout has been performed also with an automatic design tool. The target operation frequency was 24MHz. The timing for AHF to cope with the possible condition such as Park, Hold, and Sniff mode is shown in Figure 10. The sample data used in case all channels were used are shown in Table 3.

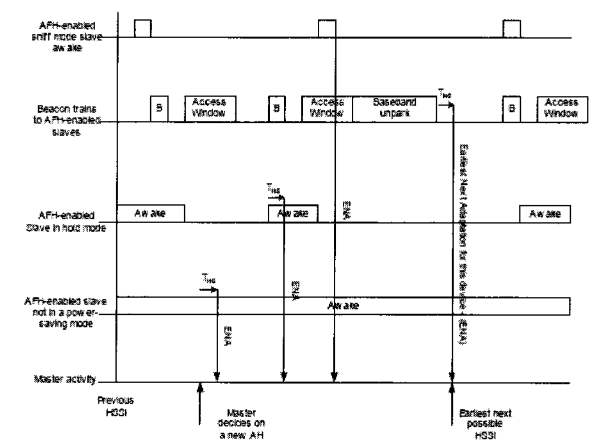


Fig 10. Timing constraints from Park, Hold and Sniff on next HSSI

Table 3. Sample data set applied when all channels were used

Hop Sequence [k] for CONNECTION STATE:
CLK start: 0x0000010
ULAP: 0x00000000
Used Channels: 0x7fffffffffffffffffff

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	08 08	10 10	12 12	14 14	16 16	18 18	20 20	22 22
0x0000030	24 24	26 26	28 28	30 30	32 32	34 34	36 36	38 38
0x0000050	40 40	42 42	44 44	46 46	48 48	50 50	52 52	54 54
0x0000070	56 56	58 58	60 60	62 62	32 32	36 36	34 34	38 38
0x0000090	40 40	44 44	42 42	46 46	48 48	52 52	50 50	54 54
0x00000b0	56 56	60 60	58 58	62 62	64 64	68 68	66 66	70 70
0x00000d0	72 72	76 76	74 74	78 78	01 01	05 05	03 03	07 07
0x00000f0	09 09	13 13	11 11	15 15	64 64	66 66	68 68	70 70
0x0000110	01 01	03 03	05 05	07 07	72 72	74 74	76 76	78 78
0x0000130	09 09	11 11	13 13	15 15	17 17	19 19	21 21	23 23
0x0000150	33 33	35 35	37 37	39 39	25 25	27 27	29 29	31 31
0x0000170	41 41	43 43	45 45	47 47	17 17	21 21	19 19	23 23
0x0000190	33 33	37 37	35 35	39 39	25 25	29 29	27 27	31 31
0x00001b0	41 41	45 45	43 43	47 47	49 49	53 53	51 51	55 55
0x00001d0	65 65	69 69	67 67	71 71	57 57	61 61	59 59	63 63
0x00001f0	73 73	77 77	75 75	00 00	49 49	51 51	57 57	59 59
0x0000210	53 53	55 55	61 61	63 63	65 65	67 67	73 73	75 75
0x0000230	69 69	71 71	77 77	00 00	02 02	04 04	10 10	12 12
0x0000250	06 06	08 08	14 14	16 16	18 18	20 20	26 26	28 28
0x0000270	22 22	24 24	30 30	32 32	02 02	06 06	10 10	14 14
0x0000290	04 04	08 08	12 12	16 16	18 18	22 22	26 26	30 30
0x00002b0	20 20	24 24	28 28	32 32	34 34	38 38	42 42	46 46
0x00002d0	36 36	40 40	44 44	48 48	50 50	54 54	58 58	62 62
0x00002f0	52 52	56 56	60 60	64 64	34 34	36 36	50 50	52 52
0x0000310	38 38	40 40	54 54	56 56	42 42	44 44	58 58	60 60
0x0000330	46 46	48 48	62 62	64 64	66 66	68 68	03 03	05 05
0x0000350	70 70	72 72	07 07	09 09	74 74	76 76	11 11	13 13
0x0000370	78 78	01 01	15 15	17 17	66 66	70 70	03 03	07 07
0x0000390	68 68	72 72	05 05	09 09	74 74	78 78	11 11	15 15
0x00003b0	76 76	01 01	13 13	17 17	19 19	23 23	35 35	39 39
0x00003d0	21 21	25 25	37 37	41 41	27 27	31 31	43 43	47 47
0x00003f0	29 29	33 33	45 45	49 49	19 19	21 21	23 23	25 25

V. CONCLUSION

If there is a third wireless device when two Bluetooth devices are trying to make communications with each other using a shared frequency within the shared range, which is called a piconet, an interference problem might occur when the third wireless devices hops into the shared frequency. In this paper, we proposed a software-based adaptive frequency hopping method so that more than two wireless devices can stay connected without frequency collision. We proposed a way of implementing an adaptive frequency hopping method combining software and hardware design efficiently. The proposed method was described with HDL and automatically laid out. Test chips were fabricated and showed avoiding interference operating at 24MHz clock at room temperature. With this contribution, we can provide a way to solve the problem of frequency collision problems in the real-time communications between handheld PCs or PDAs.

It must be noted, however, that the proposed adaptive frequency hopping method can be used more as a means to improve the performance of a Bluetooth piconet in the presence of other non-hopping systems in the 2.4-GHz ISM band than a way to improve the performance among coexisting Bluetooth piconets.

REFERENCES

[1] http://www.ericson.com/bluetooth, "Adaptive Frequency Hopping for Reduced Interference between Bluetooth and Wireless LAN," white paper, 2003.

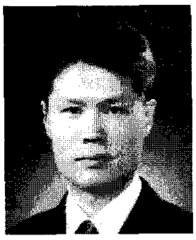
[2] Cheol-Hee Park, et. al., "Coexistence mechanism based on adaptive frequency hopping for interference-limited WPAN applications," Signal Processing and Its Applications, 2003. Proceedings. Seventh International Symposium on , Volume: 1 , 1-4 July 2003 pp. 269~272 vol.1

[3] Bin Zhen, et. al., "The analysis of coexistence mechanisms of Bluetooth," Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th , Volume: 1 , 6-9 May 2002 pp. 419~423 vol.1

[4] Das, A. et al., "Adaptive link-level error recovery mechanisms in Bluetooth," Personal Wireless Communications, 2000 IEEE International Conference on , 17-20 Dec. 2000 pp. :85~89

[5] Bluetooth Special Interest Group, "Specifications of the Bluetooth System, vol. 1, v.1.0B 'Core' and vol. 2 v1.0B 'Profiles'," December 1999.

[6] IEEE Std. 802-15 Task Group on Coexistence, "Draft Recommended Practice for Information Technology, Part 15.2: Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in the Unlicensed Frequency Bands," March 2003.



Sangook Moon was born in Korea, in 1971. He received the B.S., M.S. and Ph.D. degree in electronic engineering from Yonsei University, Korea in 1995, 1997, and 2002 respectively. After the graduation, he had been working at Hynix Semiconductor as a senior engineer. In 2004, he joined the department of Electronic and Information Security Engineering at Mokwon University, where he is currently an assistant professor. His current research interests include VLSI, crypto-processors, computer arithmetic, u-healthcare, and embedded SoCs