Building Gateway Interconnected Heterogeneous ZigBee and WiFi Network Based on Software Defined Radio

Shuhao Wang, Yonggang Li(✉), Chunqiang Ming, and Zhizhong Zhang

School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
lyg@cqupt.edu.cn

Abstract. The ZigBee Alliance Lab proposes the concept of ZigBee-WiFi network. ZigBee-WiFi network has a broad development space when combined with the advantages of ZigBee and WiFi. However, since ZigBee and WiFi are heterogeneous in various aspects, it is necessary to find a way to interconnect the two networks. The traditional approach is to design dedicated hardware. Since the physical layer functions and part of MAC layer functions in the hardware are fixed, this method cannot adapt to the new physical layer and signal processing algorithms. Software Defined Radio (SDR) is an emerging and flexible method of transferring signal processing components from dedicated hardware to a combination of software and general purpose processors. In this paper, we use SDR in conjunction with the Universal Software Radio Peripheral (USRP) to build a flexible and universal ZigBee-WiFi gateway for interconnecting heterogeneous ZigBee and WiFi networks. The gateway has the ability to simultaneously receive and demodulate ZigBee packets, create and transmit WiFi data frames. A comprehensive performance test confirmed that the built gateway can well interconnect heterogeneous ZigBee and WiFi networks. And the built gateway provides a reference prototype for the interconnection research of heterogeneous networks.

Keywords: Software Defined Radio · ZigBee-WiFi gateway · Heterogeneous network · USRP

1 Introduction

ZigBee is an open wireless standard designed to provide the foundation for the Internet of Things (IoT) by enabling items to work together. ZigBee is often chosen as the technology to connect things because of its network flexibility, interoperability and low power consumption [1]. WiFi functions as a bridge between network base stations and a large number of portable terminal devices. However, ZigBee and WiFi are heterogeneous in various aspects. WiFi and ZigBee have different transmit power, asynchronous time slots and incompatible physical layers. ZigBee devices and WiFi devices cannot communicate directly, so it is necessary to find a way to connect these two heterogeneous networks.

H. Gao et al. (Eds.): ChinaCom 2019, LNICST 312, pp. 445–456, 2020.
https://doi.org/10.1007/978-3-030-41114-5_33

For this limitation, The traditional approach is to design dedicated hardware based on heterogeneous networks. The physical layer function and some MAC layer functions are integrated in the hardware circuit. In [2], The author designed the interface between ZigBee and the WiFi communication standard. In order to realize the conversion of ZigBee protocol data and WiFi protocol data frames. Smart Home Automation communicates through a dedicated processor. It sends commands to the ZigBee coordinator and the devices connected to the WiFi network [3]. This approach lacks flexibility. It cannot cope with new physical layers and signal processing algorithms. But software defined radios overcome this drawback.

Software Defined Radio (SDR) is a type of radio communication system where some or all signal processing components are implemented in software and executed on a General Purpose Processor (GPP) [4]. GNU Radio is an open source software framework for efficient deployment of SDR applications [5], which is used in conjunction with Universal Software Radio Peripheral (USRP). For the new protocol standard IEEE 802.11p, the IEEE 802.11p transceiver is implemented with GNU Radio and USRP [6]. IPTS algorithm in OFDM system is improved to reduce computational complexity based on GNU Radio platform [7]. In [8], the authors use the advantages of GNU Radio and the USRP platform to verify the performance of unsampled WiFi in decoding performance and energy efficiency. The POLYPHONY prototype can be implemented via GNU Radio and deployed to a 16-node enterprise network [9].

In this paper, we use a flexible way to interconnect heterogeneous networks. SDR is used to build a universal gateway to interconnect heterogeneous ZigBee and WiFi networks. We implement the ZigBee-WiFi gateway based on GNU Radio and exe-cute it on the USRP N210. The gateway first receives and demodulates ZigBee packets. An IEEE 802.11 data frame is then created based on the extracted data payload. Finally, the physical layer is modulated and transmitted using OFDM technology. Through three experiments, it was confirmed that the gateway has good performance. The built gateway can well interconnect ZigBee and WiFi networks. The rest of the paper is organized as follows. In Sect. 2 we introduce the GNU Radio framework and the ZigBee-WiFi gateway implementation over GNU Radio. The gateway test platform is established in Sect. 3. Performance measures and numerical results, as well as discussion, are given in Sect. 4. Finally, we conclude this paper in Sect. 5.

2 System Design

2.1 GNU Radio and Flow Graph Construction

GNU Radio is an open source software framework for rapid deployment of SDR applications. We implement the ZigBee-WiFi gateway based on GNU Radio and execute it on the USRP N210. The gateway is capable of interconnecting heterogeneous ZigBee and WiFi networks. The ZigBee-WiFi gateway flow graph is shown in Fig. 1. The development languages used by GNU Radio are Python and C. Signal processing blocks in GNU Radio are generally written in C, such as the block IIR Filter, FFT.

The gateway flow graph can be divided into two parts in Fig. 1. The left part of the block diagram represents the ZigBee packet reception, and the other side represents the creation of the WiFi data frame. In GNU Radio companion, a set of signal processing

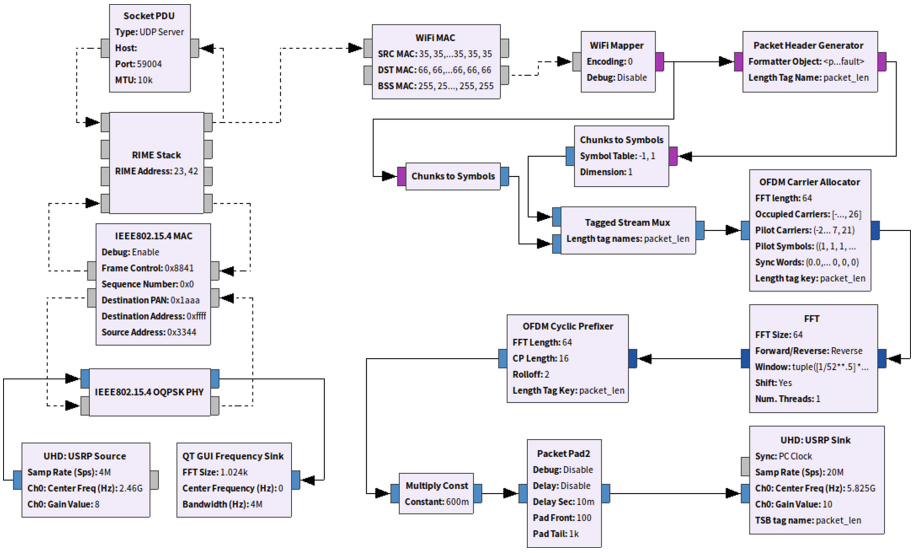

Fig. 1. Overview of the ZigBee-WiFi gateway flow graph in GNU Radio companion

blocks are connected in a specified order to generate a gateway flow graph. Python generates scripts based on flow graph to create a complete signal processing flow. The USRP is the hardware peripheral of GNU Radio. GNU Radio calls the USRP through the UHD block, and the two combined to build a radio communication system. The USRP consists of RF daughter boards, antennas, ADCs, DACs, FPGAs, etc., to receive and transmit radio signals.

There are two ways to transfer GNU Radio data, namely Stream tagging and Message passing. The tag stream is synchronized with the data stream and is used to store metadata and control information. GNU Radio typically sends and receives packets based on packets, introducing asynchronous messaging. In Fig. 1, the blue port and the solid line connection indicate synchronous transmission, and the gray port and dashed line connection indicate asynchronous transmission.

2.2 ZigBee Packet Reception

In Fig. 1, the left portion of the block diagram of the gateway flow graph represents the ZigBee packet reception. In the receive and demodulation process of the data packet, the direction of data information flows first from the UHD USRP Source block to the OQPSK PHY. The physical layer demodulates the physical layer protocol data unit (PPDU), and the extracted payload is delivered to the MAC block. The MAC layer parses the MAC layer data frame and passes the payload to the upper RIME Stack. Finally, the RIME Stack block parses and restores the valid data payload in the ZigBee packet. The following briefly describes the functions of each block in receiving ZigBee packets.

The physical layer in Fig. 1 is encapsulated in a hierarchical block. It contains a complete flow graph that hides the details of signal processing. The OQPSK PHY block

implementation is based on the UCLA ZigBee PHY. In Fig. 2, the MAC data frame is passed to the physical layer as a payload, called PSDU. The action receiver of the SHR implements symbol synchronization, and PHR represents the length of the payload. The SHR, PHR, and PHY payloads together form a PHY packet PPDU. The physical channel bandwidth is 5 MHz, and the center frequency of each channel is as follows

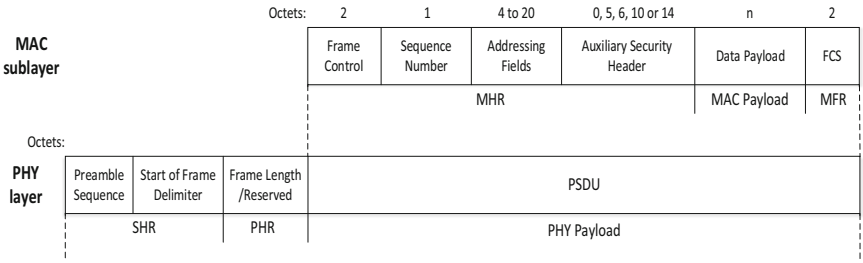

Fig. 2. Structure of the IEEE 802.15.4 physical layer protocol data unit (PPDU)

$$f_c = 2405 + 5(k - 11)11 \leq \mathrm{k} \leq 26 \tag{1}$$

The physical layer uses OQPSK modulation and demodulation technology. The general OQPSK modulation signal is expressed as

$$s(t) = \sqrt{p}c_1(t)\cos[\omega_0 t + \theta_d(t)] + \sqrt{p}c_2(t)\sin[\omega_0 t + \theta_d(t)] \tag{2}$$

where $\theta_d(t)$ represents data phase modulation, $c_1(t)$ and $c_2(t)$ represent mutually independent orthogonal spread codes.

The USRP source block is connected to the hardware USRP to transmit the received data frame information to the OQPSK PHY block. When the OQPSK PHY block demodulates the physical layer data frame, the OQPSK signal in the complex baseband can be expressed as

$$s(k) = \sqrt{E_b}e^{j(\omega(k)kT_s+\theta_0)}\left[\sum_i a_i g(kT_s - iT) + j\sum_i b_i g(kT_s - iT) + j\right] \tag{3}$$

where E_b represents the average bit energy, T is the symbol period, T_S is the signal sampling period, and g represents the impulse response of the pulse filter. a_i and b_i represent orthogonal modulation I and Q channel respectively. The OQPSK PHY block parses the PPDU and extracts the payload to the MAC layer.

The MAC block can receive the data packet from the upper layer as the data payload, and add the frame header MHR and the frame tail MFR to form a complete MPDU. When receiving data from the following physical layer, the MAC layer removes the header of the MPDU. Then check the CRC in the end of the frame, and if it is correct, extract the payload to the upper layer. Currently, the MAC layer in Fig. 1 has basic encapsulation framing and parsing capabilities. The Rime stack is a lightweight network stack for WSN with low power consumption and simple implementation [10]. The Rime stack in Fig. 1 is only a stack of frame transmission and frame reception. It is equivalent to the

network layer in a layered architecture. It adds header information to the upper layer data packets, enabling broadcast communication, unicast communication, and reliable unicast communication. The address of the rime stack is configured in GNU Radio companion. In terms of frame reception, it parses data from the MAC layer and extracts the payload.

2.3 WiFi Data Frame Transmission

The right side of the block diagram in Fig. 1 represents the generation of an IEEE 802.11 data frame PPDU. The physical layer of IEEE 802.11a uses OFDM technology and operates in the 5 GHz band. It can support channel bandwidths of 20 MHz, 10 MHz, and 5 MHz. OFDM is a special multi-carrier modulation technique that modulates serial data streams in parallel on multiple orthogonal subcarriers.

The functions of block are described below. The WiFi MAC block generates a MAC data frame based on the payload of the ZigBee packet, which will serve as the data payload for the physical layer. The WiFi Mapper creates the rate and service fields of the PLCP header of the physical layer frame PPDU. Next, the Packet Header Generator block is used to generate the header of the physical layer frame PPDU, which consists of a PLCP Header and a PLCP Preamble. The next block Chunks to Symbols modulates the signal. The OFDM Carrier Allocator block acts to create OFDM symbols, meaning that the incoming complex streams are allocated to different data subcarriers. An OFDM symbol has a plurality of orthogonal subcarriers, the starting time is t_s, and the OFDM symbol at time t can be expressed as

$$s(t) = \begin{cases} \mathrm{Re}\left\{ \sum_{i=0}^{N-1} d_i rect(t - t_s - \frac{T}{2}) \exp[j2\pi f_i (t - t_s)] \right\} & \mathrm{t}_s \le \mathrm{t} \le \mathrm{t}_s + \mathrm{T} \\ 0 & \mathrm{t} < \mathrm{t}_s \wedge t > \mathrm{t}_s + \mathrm{T} \end{cases} \tag{4}$$

where T is an OFDM symbol period, N represents the number of subcarriers, d_i is modulation data on a corresponding subcarrier, and f_i is a frequency of a corresponding subcarrier. Since each subcarrier contains an integer number of periods in the OFDM symbol period, and the number of adjacent subcarrier periods differs by 1, the subcarriers are orthogonal.

$$\frac{1}{T} \int_0^T \exp(j\omega_n t) \times \exp(-j\omega_m t) dt = \begin{cases} 1 & m = n \\ 0 & m \ne n \end{cases} \tag{5}$$

For the integral operation of the corresponding subcarriers in Eq. (4), within the time interval T, the following equation can be derived

$$\begin{aligned} \hat{d}_j &= \frac{1}{T} \int_{t_s}^{t_s+T} \exp\left(-j2\pi \frac{j}{T}(t - t_s) \right) \sum\nolimits_{i=0}^{N-1} d_i \exp\left(j2\pi \frac{j}{T}(t - t_s) \right) d_t \\ &= \sum\nolimits_{i=0}^{N-1} d_i \int_{t_s}^{t_s+T} \exp\left(j2\pi \frac{i - j}{T}(t - t_s) \right) d_i = d_j \end{aligned} \tag{6}$$

Obtained by Eq. (6), coherent demodulation of the j-subcarrier can obtain the expected symbol d_j. Since the other subcarriers are integrated in the integration interval, the result

is zero. Based on the orthogonal characteristics of subcarriers, the modulation of OFDM can be obtained by IDFT operation to obtain a discretized time domain signal, and the sampling rate of s(t) is set to $T \Big/ N$

$$s_k = s(kT \Big/ N) = \sum_{i=0}^{N-1} d_i \exp j\left(\frac{2\pi ik}{N}\right) \; 0 \le k \le N-1 \tag{7}$$

The FFT block uses IFFF operations to convert frequency domain data into time domain data, which is more efficient and faster than IDFT. In addition, the OFDM Cyclic Prefixer block adds a cyclic prefix to the OFDM symbol. In order to solve the Inter Symbol interference (ISI) caused by the multipath delay of the wireless channel transmission. The last block, the USRP Sink block, uses the hardware USRP to convert IEEE 802.11 data frames into analog signals and move them to the intermediate frequency. The radio board is transmitted at a frequency of 5.825 GHz.

3 Experimental Environment Construction

The ZigBee-WiFi gateway flow graph has been built in GNU Radio companion, and then the flow graph needs to be executed to measure the performance of the ZigBee-WiFi gateway. The USRP acts as a hardware peripheral for the ZigBee-WiFi gateway flow graph, and the two are combined to implement the corresponding functions of the ZigBee-WiFi gateway. Currently the gateway is unidirectional. It receives and demodulates the ZigBee packet, then extracts the payload and encapsulates it to form a WiFi data frame for transmission. The gateway cannot receive and demodulate WiFi packets and then generate ZigBee packets. The test scenario is shown in Fig. 3.

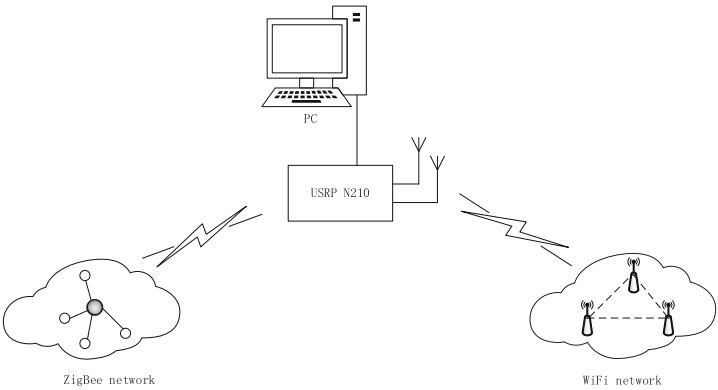

Fig. 3. ZigBee-WiFi gateway performance test scenario

The ZigBee device selects the TelosB node, and the USRP N210 daughter board UBX-40 acts as the RF front end. It has a full-duplex transceiver with tunable frequencies from 10 MHz to 6 GHz. Therefore, the USRP can simultaneously receive ZigBee packets and send WiFi data frames. Table 1 shows the important components of the experimental test.

Table 1. The important components of the experimental test.

Component	Detailed information
ZigBee node	Type TelosB
USRP	Type N210
Daughterboard	UBX-40
GNU Radio	Version 3.7.10
UHD	Version 3.10.1
Operating system	Ubuntu 16.04
CPU	I5-7500 3.4 GHz
Memory	8 G

We present details of the important components of the experimental scenario in Table 1. In order to simplify the test, there is one node device in each of the ZigBee network and the WiFi network. According to the literature [11], a USRP N210 is configured as a WiFi receiver in the WiFi network to conveniently view the received data frames. Below we conducted three experiments to measure the performance of the built ZigBee-WiFi gateway. The first experimental test gateway receives and demodulates ZigBee packets. The second experiment simulates the reception rate of the WiFi packet when the gateway sends the WiFi data frame through the channel containing the noise. The third experimental test gateway receives and demodulates the ZigBee data packet and then transmits the WiFi data frame in the real scene.

4 Performance Measures and Numerical Results

The performance of the ZigBee-WiFi gateway is mainly reflected in two aspects, one is to receive and demodulate ZigBee data packets. On the other hand, the ability to create IEEE 802.11 (WiFi) data frames and transmit. Next, The first experiment tests that the gateway receives ZigBee packets. We set the ZigBee node TelosB to operate at 2.46 GHz and select 22 channels. The transmitted ZigBee packet sizes are 28 bytes, 48 bytes, 68 bytes, and 88 bytes, respectively. The sending interval is 250 ms, and the number of packets is increased from 80 to 560. The receiving gain of the USRP is set to a minimum of 0 dB. The experimental results are shown in Fig. 4.

The ZigBee nodes can only communicate on the same channel, so the USRP also works at 2.46 GHz. In the gateway flow graph, we connect the wireshark block to the output port of the OQPSK PHY block. And then the file sink block connect to the output port of the wireshark block. The file sink block generates a PCAP format file to view the number of received ZigBee packets. In Fig. 4, As the size of the packet increases, the reception rate decreases to some extent. When 480 ZigBee packets are transmitted, the reception rate of the 28-byte packet and the 88-byte packet is 98.9% and 93.3%, respectively. When the USRP receive gain is increased to 6 dB, the gateway's receive rate for the 88 Byte ZigBee packet can reaches 100%. This can indicate that the gateway has good ability to receive and demodulate ZigBee packets.

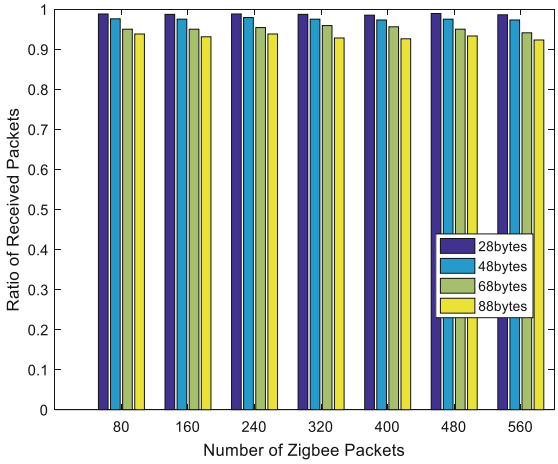

Fig. 4. ZigBee packet reception rate of gateways under different packet sizes

The second test simulates the gateway flow graph in GNU Radio companion. First, the flow graph will create WiFi data frame, then transmit through the channel containing noise, and finally measure the WiFi packet reception rate. We compare the WiFi packet reception rate under different modulation mode when the channel contains Gaussian noise and uniform noise respectively. The simulation results are shown in Figs. 5 and 6.

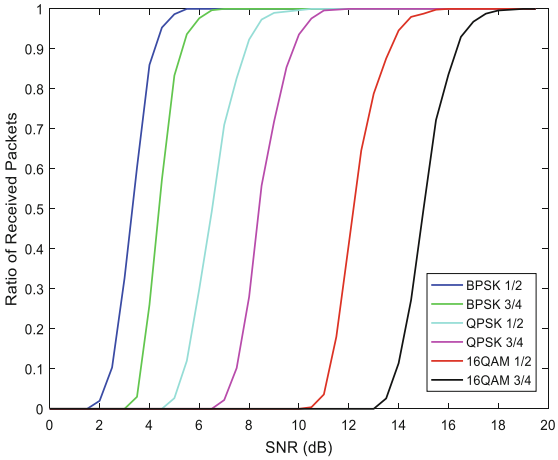

Fig. 5. Simulation of WiFi packet reception rate over the channel containing Gaussian noise

In the simulation, the created WiFi data frame length is 114 bytes. The operating frequency of the WiFi is set to 5.825 GHz and the bandwidth is 20 MHz. In addition, SNR starts from 0 dB and increases by 0.5 dB each time until 20 dB. The gateway generates 100 WiFi data frames each time and then transmits them through a channel containing noise. The WiFi packet reception rate is obtained by checking the number of

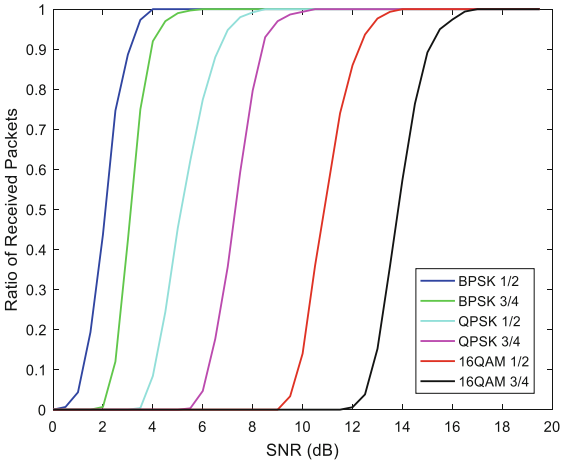

Fig. 6. Simulation of WiFi packet reception rate over the channel containing uniform noise

the received WiFi packets. In Fig. 5, the WiFi packet is transmitted through a channel model containing Gaussian noise. For any modulation mode, the packet reception rate increases as the SNR increases.

The simulation curves in Figs. 5 and 6 are reasonable because the higher order modulation mode 16QAM requires a higher SNR than QPSK and BPSK. For the same modulation scheme, such as QPSK, a higher SNR is required in a Gaussian noise channel than in a uniform noise channel to achieve the same WiFi packet reception rate.

The third experiment tests that the performance of ZigBee-WiFi gateway in the real scene. The purpose of this test is to view the real-time signal processing capabilities of the ZigBee-WiFi gateway and the transmission of WiFi data frames in the actual scene. The ZigBee packets is sent by the TelosB node. And the other USRP acts as the WiFi receiver which receives the WiFi data frame transmitted by the ZigBee-WiFi gateway. The experimental results are shown in Figs. 7 and 8. In this experiment, the size of the ZigBee packet sent is 88 bytes. The transmission interval is 250 ms. The ZigBee-WiFi gateway receives and demodulates the ZigBee packets, and then extracts the payload to generate IEEE 802.11 data frame with a length of 114 Bytes.

In Fig. 7, the USRP acts as the WiFi receiver, but it cannot record SNR values. The SNR is calculated to be equivalent to the difference between the transmit gain of the gateway and the receive gain of the WiFi receiver. The symbol S in BPSK-1/2-S in Fig. 7 represents the Gaussian channel simulation scenario of Fig. 5, and R in BPSK-1/2-R represents the actual scene. The dashed curve in Fig. 7 is also the Gaussian channel scene simulation result in Fig. 5. The results are reasonable in the sense that a higher SNR is required in the actual scenario to achieve the same receiving rate. In addition, the WiFi packet reception rate can reach 1 as the SNR increases. The experimental results in the actual scene are not much different from the simulation results. This shows that the built ZigBee-WiFi gateway has good ability to create IEEE 802.11 data frames and transmit them. Due to other WiFi device interference in the actual environment and oscillator drift in hardware USRP, the curve in the actual scene is not smooth and has some differences

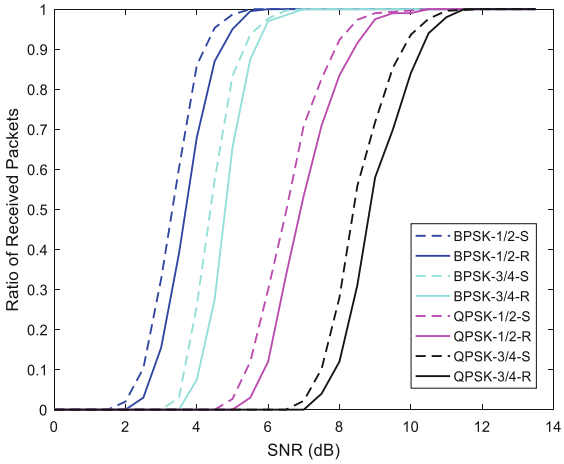

Fig. 7. Compare WiFi packet reception rate when ZigBee-WiFi gateway transmits WiFi data frame in simulated scene and real scene

from the simulation results in Fig. 7. It is more likely that nonlinearities in the amplifier may slightly interfere with the signal, resulting in packet errors.

It shows the runtime proportion of individual blocks when the ZigBee-WiFi gateway is executed in Fig. 8. The runtime proportion of all blocks is added up to a total of 1. The block runtime is measured with the help of gr-perf-monitorx in GNU Radio [12]. The five red bars indicate the runtime of the IEEE 802.15.4 OQPSK PHY internal block when the gateway demodulates the ZigBee packet. The three blue block probes work in conjunction with the tool gr-perf-monitorx to measure the block runtime. The cyan bar represents the runtime of the block that creates the WiFi data frame, and the OFDM Carrier Allocator block accounts for 74.14%. Due to the high computational complexity of the block, it is the core block for creating WiFi data frames. In Fig. 8, the various

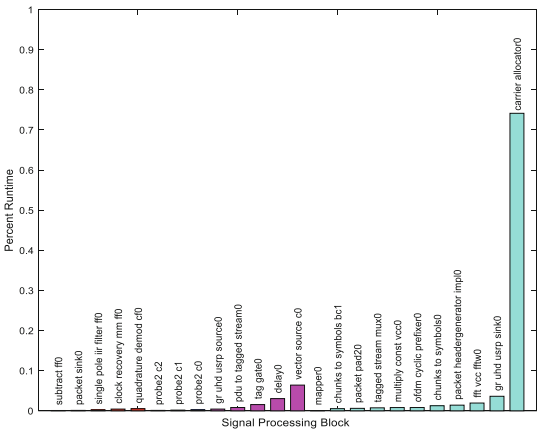

Fig. 8. Runtime proportion of individual signal processing blocks

signal processing modules of the gateway operate normally. Combined with Fig. 7, it can be concluded that the gateway has better real-time signal processing capability under actual scenarios.

5 Conclusion

In this paper, we use SDR combined with USRP to build a flexible and universal ZigBee-WiFi gateway which interconnects heterogeneous ZigBee and WiFi network. A comprehensive performance measurement of the gateway is performed. The first experiment tests that the gateway receives ZigBee packets, and the result shows that the gateway has good ability to receive and demodulate data packets. The second and third experiments are respectively tested in the simulation scenario and the actual scenario, comparing the receiving rate of the WiFi packet when the ZigBee-WiFi gateway transmits the WiFi packet. The results are reasonable in the sense that a higher SNR is required in the actual scenario to achieve the same receiving rate. Combined with the above three experimental results, it shows that the ZigBee-WiFi gateway has good ability to receive and demodulate the ZigBee packets and then extract the payload to generate IEEE 802.11 data frame. It proves that the built gateway can well interconnect heterogeneous ZigBee and WiFi networks.

References

1. Davoli, L., Belli, L., Cilfone, A., Ferrari, G.: From micro to macro IoT: challenges and solutions in the integration of IEEE 802.15.4/802.11 and sub-GHz technologies. IEEE Internet Things J. **5**(2), 784–793 (2017)
2. Nugroho, E., Sahroni, A.: ZigBee and wifi network interface on wireless sensor networks. In: 2014 Makassar International Conference on Electrical Engineering and Informatics (MICEEI), pp. 54–58. IEEE (2014)
3. Vivek, G.V., Sunil, M.P.: Enabling IOT services using WIFI-ZigBee gateway for a home automation system. In: 2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pp. 77–80. IEEE (2015)
4. Arcos, G., Ferreri, R., Richart, M., Ezzatti, P., Grampín, E.: Accelerating an IEEE 802.11 a/g/p transceiver in GNU radio. In: Proceedings of the 9th Latin America Networking Conference, pp. 13–19. ACM (2016)
5. Robert, M., Sun, Y., Goodwin, T., Turner, H., Reed, J.H., White, J.: Software frameworks for SDR. Proc. IEEE **103**, 452–475 (2015)
6. Bloessl, B., Segata, M., Sommer, C., Dressler, F.: Performance assessment of IEEE 802.11 p with an open source SDR-based prototype. IEEE Trans. Mob. Comput. **17**(5), 1162–1175 (2017)
7. Ming, A., Xiaosong, Z.: Improved IPTS algorithm in OFDM system based on GNU radio. In: 2018 10th International Conference on Communication Software and Networks (ICCSN), pp. 352–356. IEEE (2018)
8. Wang, W., Chen, Y., Wang, L., Zhang, Q.: From rateless to sampleless: Wi-Fi connectivity made energy efficient. In: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9. IEEE (2016)
9. Yang, P., Yan, Y., Li, X.Y., Zhang, Y.: POLYPHONY: scheduling-free cooperative signal recovery in enterprise wireless networks. IEEE Trans. Mob. Comput. **16**(9), 2599–2610 (2016)

10. Dunkels, A., Österlind, F., He, Z.: An adaptive communication architecture for wireless sensor networks. In: Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, pp. 335–349. ACM (2007)
11. Bloessl, B., Segata, M., Sommer, C., Dressler, F.: An IEEE 802.11 a/g/p OFDM receiver for GNU radio. In: Proceedings of the Second Workshop on Software Radio Implementation Forum, pp. 9–16. ACM (2013)
12. Rondeau, T.W., O'Shea, T., Goergen, N.: Inspecting GNU radio applications with controlport and performance counters. In: Proceedings of the Second Workshop on Software Radio Implementation Forum, pp. 65–70. ACM (2013)